

2

DTIC FILE COPY

NAVAL POSTGRADUATE SCHOOL Monterey, California

AD-A219 965



THESIS

DTIC
ELECTE
APR 02 1990
S E D

ADAPTIVE CONTROL METHODS FOR MECHANICAL
MANIPULATORS: A COMPARATIVE STUDY

by

Hamadi Jamali

December 1989

Thesis Advisor

Roberto Cristi

Approved for public release; distribution is unlimited.

90 04 02 163

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE				
4 PERFORMING ORGANIZATION REPORT NUMBER(S)			5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b OFFICE SYMBOL (If applicable) 62	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000			7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	
8a NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)			10 SOURCE OF FUNDING NUMBERS	
			PROGRAM ELEMENT NO	PROJECT NO.
			TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) ADAPTIVE CONTROL METHODS FOR MECHANICAL MANIPULATORS: A COMPARATIVE STUDY / UNCLASSIFIED				
12 PERSONAL AUTHOR(S) Hamadi Jamali				
13a TYPE OF REPORT Master's Thesis		13b TIME COVERED FROM TO	14 DATE OF REPORT (Year, Month, Day) December 1989	15 PAGE COUNT 164
16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Automatic Control Systems, Identification, Adaptive Control and Robotics.	
FIELD	GROUP	SUB-GROUP		
19 ABSTRACT (Continue on reverse if necessary and identify by block number) A new adaptive control law for mechanical manipulators that maintains uniformly good performance over a wide range of motions and payloads is developed. This control strategy combines properties from both the Model Reference Adaptive Control and the Self Tuning Regulator Theory and serves to extend the Adaptive Model Following Control approach into using a nonlinear reference model. The design procedure is simple resulting in an overall system which is globally stable and offers itself to microcomputer implementation. The effectiveness of the approach is demonstrated on several computer simulations which compares its performances against some of the commonly known adaptive control techniques. Also presented is a comparison of the computation complexity of different methods used in deriving the dynamic equations of motion of a mechanical manipulator as well as a survey of various robot control methodologies available in the literature today.				
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Dr. Roberto Cristi			22b TELEPHONE (Include Area Code)	22c OFFICE SYMBOL 62

Approved for public release; distribution is unlimited

Adaptive Control Methods for Mechanical Manipulators:
A Comparative Study

by

Hamadi Jamali
LTJG, Marine Royale
B.S., Lycee El Malqui, Rabat, 1980
Ing. Mech., Ecole Royale Navale, Casablanca, 1984
Ing. Mech. Sp., C.I.N., Saint Mandrier, France, 1985

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

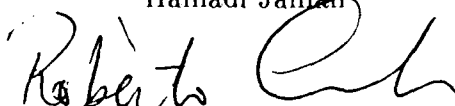
December 1989

Author:

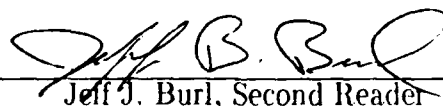


Hamadi Jamali


Approved by:



Roberto Cristi, Thesis Advisor



Jeff J. Burl, Second Reader



John P. Powers, Chairman,
Department of Electrical and
Computer Engineering

ABSTRACT

A new adaptive control law for mechanical manipulators that maintains uniformly good performance over a wide range of motions and payloads is developed. This control strategy combines properties from both the Model Reference Adaptive Control and the Self Tuning Regulator Theory and serves to extend the Adaptive Model Following Control approach into using a nonlinear reference model.

The design procedure is simple resulting in an overall system which is globally stable and offers itself to microcomputer implementation. The effectiveness of the approach is demonstrated on several computer simulations which compares its performances against some of the commonly known adaptive control techniques.

Also presented is a comparison of the computation complexity of different methods used in deriving the dynamic equations of motion of a mechanical manipulator as well as a survey of various robot control methodologies available in the literature today.

Accession For	
NTIS GFA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



TABLE OF CONTENTS

I. ROBOTICS: AN OVERVIEW	1
A. INTRODUCTION	1
1. Continuous process controls	1
2. Hard automation	1
3. Flexible automation	1
B. A MECHANICAL MANIPULATOR:	
DEFINITION AND CHARACTERISTICS	2
C. ROBOTICS APPLICATIONS	5
D. STRUCTURE OF A TYPICAL ROBOT SYSTEM	5
1. The mechanical manipulator	7
2. The environment	7
3. The task	7
4. The controller	8
E. THE ROBOT CONTROL PROBLEM	9
F. LITERATURE REVIEW	10
1. Open loop control systems	11
2. Closed loop control systems	11
a. The servomechanism approaches	11
b. Adaptive techniques	13
G. THESIS OUTLINE	19
II. MECHANICAL MANIPULATOR DYNAMICS	20
A. INTRODUCTION	20

B. CLOSED-FORM LAGRANGIAN FORMULATION	
OF MANIPULATOR DYNAMICS	21
C. RECURSIVE LAGRANGIAN MANIPULATOR DYNAMICS	30
D. RECURSIVE NEWTON-EULER'S MANIPULATOR DYNAMICS	36
E. CONCLUSIONS	41
III. ADAPTIVE CONTROL FOR MECHANICAL MANIPULATORS	44
A. INTRODUCTION	44
B. MECHANICAL MANIPULATOR DYNAMICS	
IN TERM OF STATE VARIABLES	45
C. SIMULATION STUDY OF SOME ADAPTIVE CONTROL METHODS	48
1. Computed torque technique	48
2. Variable structure systems	62
3. Adaptive linear model following control	63
4. Adaptive perturbation control	74
D. ADAPTIVE NONLINEAR MODEL FOLLOWING CONTROL	89
E. CONCLUSIONS AND FUTURE RESEARCH	146
APPENDIX : A TWO LINK MECHANICAL MANIPULATOR	
DYNAMIC EQUATIONS	147
LIST OF REFERENCES	152
INITIAL DISTRIBUTION LIST	157

I. ROBOTICS : AN OVERVIEW

A. INTRODUCTION

The 1980's may easily be characterized as the robotics era. The last five or six years have experienced very strong social and economical demands for advanced automation in a fast expanding domain of applications ranging from the well established car-making industry to unmanned underwater workstations [1]. Moreover, there is a widespread feeling that it is likely that robots, in the years ahead, will become crucial agents of industrial change, transforming production processes and affecting everyday lives [2]. Confronted with these facts, we are led to wonder about the reasons behind experts attaching such heavy weight to robotics and about the characteristics that make the industrial robot such a powerful tool.

An answer to these questions may be found by tracing the origins that tie robotics to automation. Roughly speaking, we can identify three types of automation [3].

1. Continuous process controls

This type of automation is used in oil refineries. It employs mostly computers with no, or little, human intervention. This type is highly automated.

2. Hard automation

Hard automation uses mainly transfer conveyor methods to handle the high volume mass production. This type of automation is based on setting up specific assembly lines with special tools and gadgets. This implies that hard automation relies on hardware which cannot be easily changed, should a change in the design of a product be called for.

3. Flexible automation

Flexible automation handles low volume batch production. Because this type aims at overcoming the limitations of hard automation, it is also referred to as programmable

automation. This kind of "machine" is designed to be flexible and to be able to react to its environment in an adaptive fashion. This is clearly different from the conventional machine which can be used only for well-defined, specialized, and preappointed tasks. The mechanical manipulators used in industrial applications fall into this category. However, even though a considerable progress has been made in introducing robots into industrial situations, there is still more to learn both in overall concepts and practicalities before a robot having a performance comparable to humans can be built [4].

The next section will describe what is meant by a "mechanical manipulator", in general, and will outline some of its characteristics.

B. A MECHANICAL MANIPULATOR: DEFINITION AND CHARACTERISTICS

A robot is a computer-controlled mechanical device that can be programmed to automatically move objects through different configurations in space. Robots are normally constructed as series of coupled rigid links, which together constitute what is called a kinematic chain. There are two types of kinematic chains [5]:

1. The linkage or the closed chain, where every link is connected to at least two other links in the chain.
2. The manipulator or the open chain, where some of the links are connected to only one other link. The articulated portion of most industrial robots is an open kinematic chain with some form of end effector attached to the final link.

A typical industrial robot is shown in Figure 1.1. The coupling of two adjacent links is referred to as a kinematic pair or joint. The most frequently encountered pairs in current industrial manipulators are the revolute or rotational joint and the prismatic or translation joint. These pairs are shown in Figure 1.2. The revolute and prismatic joints are single degree of freedom pairs. Any manipulator must have at least six degrees of freedom to enable it to achieve arbitrary real-world configurations. Thus, most industrial robots are constructed of exactly six links. The first three degrees of freedom are generally referred to

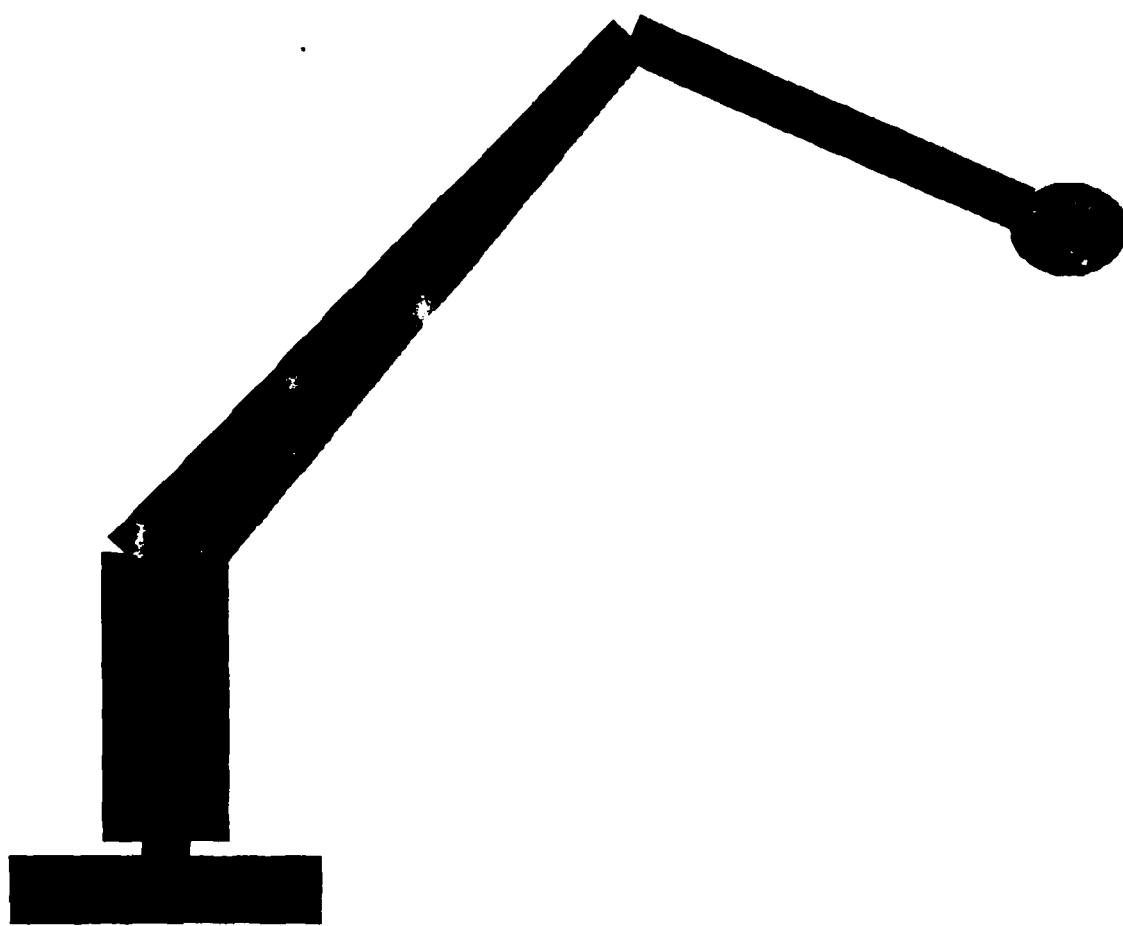


Figure 1.1: A typical industrial mechanical manipulator

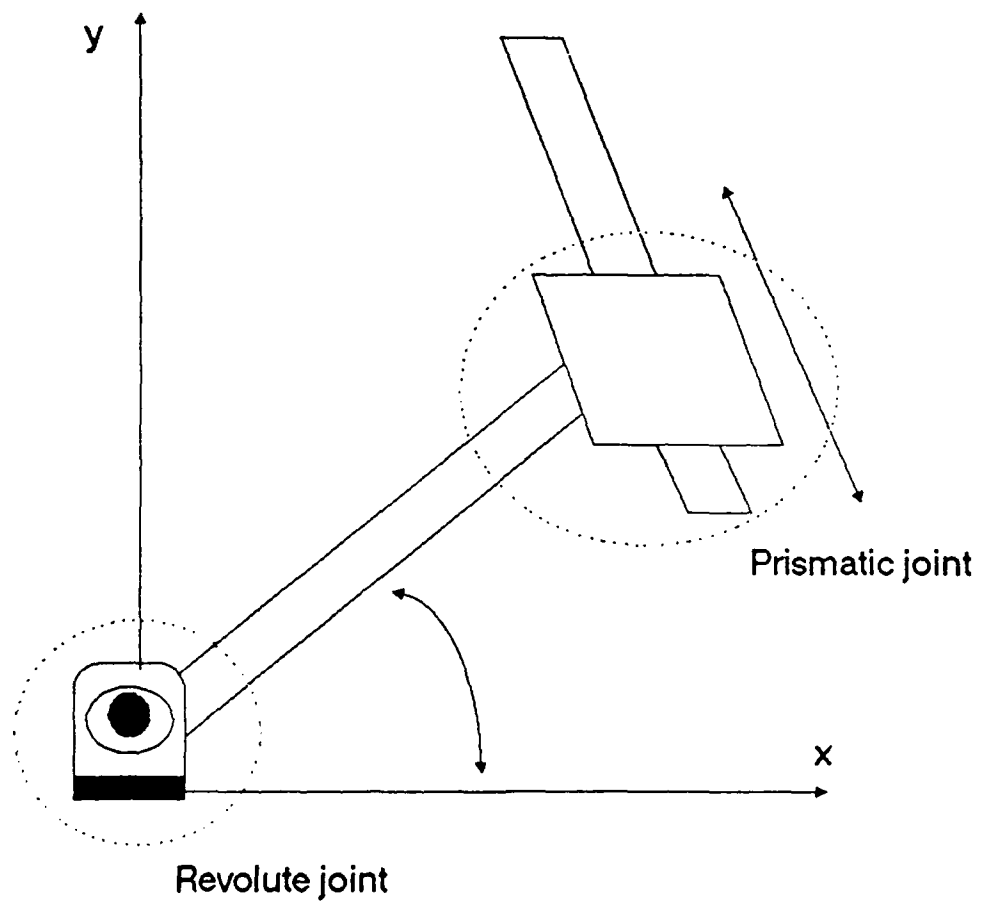


Figure 1.2: A revolute and a prismatic pairs

as an arm subassembly. They are used primarily to position the wrist unit at the workpiece. The final three degrees of freedom are referred to as the wrist subassembly, subsequently employed to orient the tool according to the configuration of the object [6]. This research is mainly concerned with the arm subassembly. The orientation of the tool will not be considered.

C. ROBOTICS APPLICATIONS

Mechanical manipulators are widely used in manufacturing and assembly tasks such as material handling, spot and arc welding, parts assembly, paint spraying, loading and unloading numerically controlled machines, and in handling hazardous materials [7]. Furthermore, it is now common belief that robot systems can be used in areas other than assembly tasks. Perhaps the most unusual application, to date, can be found in Australia, where a robot is used for sheep shearing [8]. Another application of robots is in space technology. The installation in the Space Shuttle Columbia of a remote controlled manipulator to place satellites into orbit and retrieve them when they fail is just one example [9]. Mechanical manipulators have also been used extensively in undersea research, probably even more frequently than in space; the latest example being the robotic unit used in the discovery of the Titanic [10]. Robot systems could also be used in hospitals to help paralytic people or those who must be in bed after surgery. The household robot is another dream. Military applications are also appealing. However, until all the control problems are overcome, the domain of applications of robot systems will remain limited. This will be discussed later, but, for now, a typical structure of a robot system is presented.

D. STRUCTURE OF A TYPICAL ROBOT SYSTEM

As illustrated in Figure 1.3, a robot system is, functionally, made up of four interactive parts [11]. These different parts are:

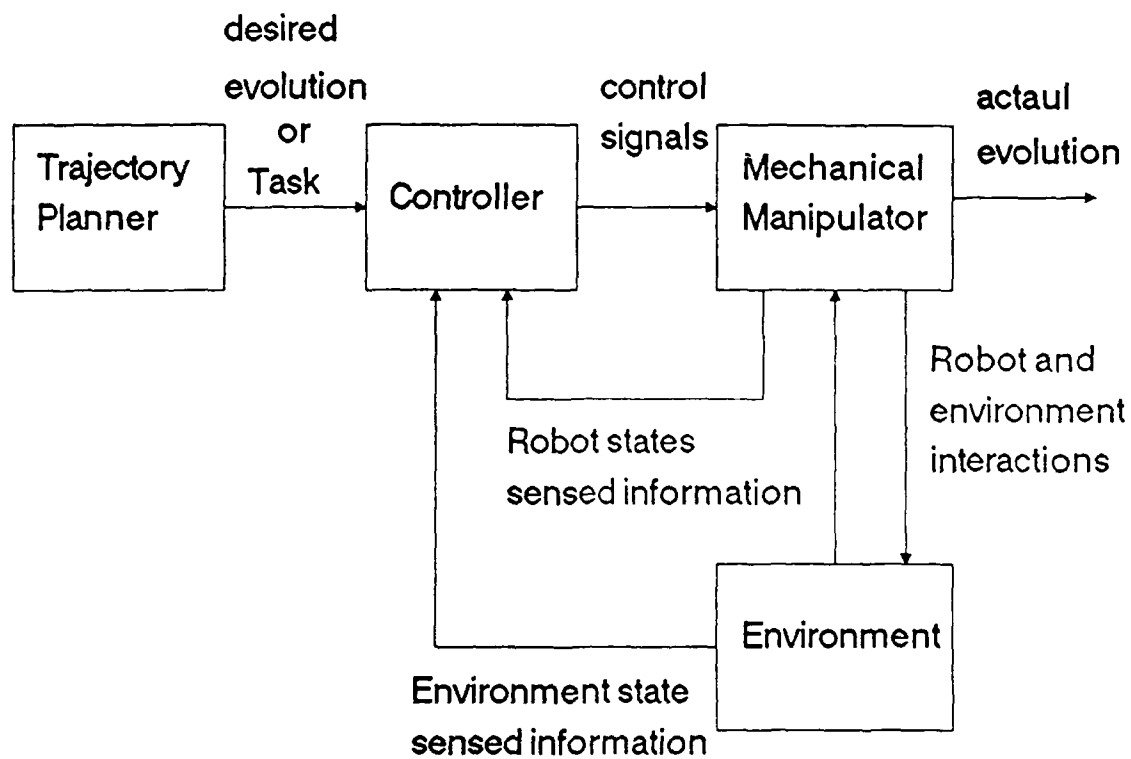


Figure 1.3: A general structure of an adaptive robot

1. The mechanical manipulator

The mechanical manipulator, itself, or the plant, is an open chain of rigid links of the type previously shown in Figure 1.1. This is the part of the machine designed to perform a specific task. Each link is powered by an actuator which physically moves the link in accordance with some prescribed control law. The joints are usually equipped with sensors to allow for the relative positions of the adjacent links to be measured.

2. The environment

The environment in which the robot operates is the physical universe surrounding the mechanical manipulator. It includes not only geometrical considerations, but also the physical laws governing this universe, the medium in which the robot is immersed and their effects on the movements of the robot. Moreover, the robot payload changes constantly either by handling parts of different masses or changing tools and configurations from one task to the other. This modifies the mass and inertia of the robot, which in turn affects its dynamic behavior. These changes must be taken into account in the formulation of the dynamic model of the mechanical manipulator.

3. The task

The task to be carried by the manipulator, or the trajectory planner, may differ from one application to another. In most cases, however, the ultimate task is driving the end effector of the manipulator to a desired position in the workspace. Naturally, this position is expressed in cartesian or world coordinates. Theoretically, the task might be accomplished in any fashion, as long as the tool reaches the final desired position and remains there. More realistically, the robot must meet certain requirements in performing its task [12]:

1. The motion must be as fast as possible, otherwise, the use of robots would not have been efficient.
2. No overshoot of the final position is allowed to prevent damages to the environment.

3. The mechanical manipulator must be able to avoid obstacles that may be present in its workspace.
4. The motion must be smooth, in order to avoid the increased wear on the mechanism and the resonances caused by vibrations due to rough and jerky motions.

Therefore, stating the initial and final conditions alone is not a sufficient task description. In most cases, it is necessary for each link to follow a prescribed trajectory in terms of position, velocity, and acceleration at each instant of time. The desired trajectory can be made by the combination of any smooth functions joining the initial and final positions and satisfying all the constraints. Cubic functions are among the most commonly used trajectories since they are easy to generate [13].

4. The controller

The controller generates the control signals that excite the corresponding actuators to produce the torques necessary to maintain a prescribed motion of the arm along the desired trajectory. The control strategy is determined according to both the control task and the mechanical manipulator equations of motion. It is, however, usually derived on the basis of a trajectory expressed in joint coordinates. Therefore, a transformation from world coordinates to link coordinates must be performed before the signals sent by the trajectory planner can be used by the controller.

In practice, the four functions described above closely interact with each other. When in operation, the computer receives, at each instant of time, information concerning the robot and information concerning the environment. By using this information in conjunction with the control law, it causes the manipulator to move toward the correct execution of the task assigned to it.

This thesis addresses the design of control systems in order for the manipulator to adapt to a changing environment, as described by functions 2 and 4 above. The next section defines in more detail the mechanical manipulator control problem and addresses some of the difficulties.

E. THE ROBOT CONTROL PROBLEM

Robotics, while bringing together many well established fields of engineering, is relatively a new science in itself. It still suffers from many unsettled points. Controlling the robot system to perform in certain way is one of its most challenging problems due to the fact that these systems are highly nonlinear. A formal statement of this problem is not, however, as difficult as trying to find a satisfying solution for it. In general terms, the robot control problem can be formulated as follows:

Given a desired trajectory generated by the trajectory planner and a mathematical model of the mechanical manipulator and its interactions with the environment, find the control algorithm which sends torque commands to the actuators in order to cause the desired motion to be realized.

One may now recognize that the robot control problem as stated here is basically a stability problem, along the given trajectory.

Mechanical manipulators may be modeled precisely enough, since their behavior is described by the known laws of mechanics [14]. This knowledge should be used in the control synthesis as extensively as possible. As stated earlier, this problem is extremely difficult because the robot systems are inherently characterized by nonlinear dynamics that include nonlinear couplings between the variables corresponding to different motions. Furthermore, the dynamic parameters of the manipulator vary with position of the joint variables, which themselves vary in time and with respect to each other. These difficulties make the implementation of real time dynamic control computationally impractical in today's computers. Therefore, one of the intriguing questions arising in the solution of the control task is to what extent one should take into account real robot dynamics in control synthesis.

Current industrial practices, in order to take advantage of the well-established linear systems and control theory, model the manipulator as a chain of constant-parameter,

uncoupled linear subsystems. These design procedures, which may be referred to as the servomechanism control methodologies, while yielding satisfactory performances at low speeds, have proven to be inefficient for faster and more accurate robotics applications [15].

Recently, more researchers have turned to adaptive control in an attempt to be able to take advantage of the full robot dynamics and to overcome the limitations of the actually available practices [16,17]. These new approaches may be referred to as adaptive control strategies for mechanical manipulators. However, no completely acceptable answer has been given to the question of how to use the knowledge of the robot dynamics to synthesize such control that would be simple enough to implement in practice and to guarantee satisfactory system behavior. Several other problems remain [18], such as:

1. The lack of adequate sensors for the acquisition and pre-processing of information received from the environment, particularly visual information.
2. The state of development of overall theory is not yet fully developed; and,
3. The slowness of the computations involved.

The first problem is more of a technological problem than theoretical one and will not be dealt with in this context. The last two problems are inherently related to the robot control problem and will constitute an important part of this research.

F. LITERATURE REVIEW

This section presents some of the most representative solutions to the mechanical manipulator control problem as of today. The main difficulty, however, in trying to review the literature, is that different approaches have been developed for different classes, types, configurations, and purposes. We will primarily consider the approaches to the control synthesis for industrial manipulators. We will also restrict ourselves to dynamic control which takes into account dynamic effects of the robotics system. Control strategies both in terms of open loop and closed loop control have been examined [19].

1. Open loop control systems

In this case, the trajectory is preplanned or prerecorded and the input torques do not depend on link position and/or velocity measurements. The performances are defined in terms of desired cost minimization criteria.

Along these lines, Kahn [20] has considered the time optimal control problem, Whitney [21] has studied the minimum energy trajectories, and Young [22] has been interested in minimizing a quadratic function in acceleration.

Due to the highly nonlinear model of the manipulator, numerical solutions only can be obtained and stored in memory. In general, this yields a control which is optimal provided the system is not affected by unexpected disturbances. Also the open loop approach leads to schemes which are very sensitive to parameter variations. Disturbance rejection and position tracking can only be achieved through accurate mechanical design. The performances of the systems are limited by the capabilities of the actuators and by the vibrations induced in the mechanism by the excitation of high frequency structural modes. The on-line implementation of such control laws is very involved and might demand a rather complex multiprocessor.

2. Closed loop control systems

These feedback control strategies are derived either through well known classical servomechanism procedures, or through more recent adaptive control techniques for their ability to account for parameter uncertainties.

a. The servomechanism approaches

Kahn and Roth [23] have proposed an approximated optimal law which, for a particular robot, has resulted in response times and trajectories reasonably close to the optimal solutions. For more complex manipulators, however, this solution might be unacceptable. Furthermore, the controller proposed in [23] is based on a bang bang approach, often unacceptable due to continuous chattering of the joint actuator's signals.

Vukobratovic and Stokic [24] addressed the more general problem of designing a controller which yields desired tracking while, at the same time, minimizes an appropriate cost criterion.

Because of analytical and computational complexity, approaches by optimal synthesis have been developed for positional control problems only. To solve the problem of tracking a prescribed trajectory, Popov and co-authors [25] introduced an alternative approach which consists of calculating, off line, the nominal trajectory by some optimal or suboptimal procedures and then following the obtained path.

The design of control systems based on the exact nonlinear model of the manipulator, in general, yields algorithms not suitable to real time implementations. For this reason, controllers based on linearized models in the neighborhood of operating conditions have been introduced. This, however, guarantees the stability of the linearized model only. Instabilities might occur in the actual system due to nonlinearities in the mechanism, coupling between different joints, or parameter variations. In order to overcome this major difficulty, several additional compensation schemes have been proposed [26,27] at the expenses of added complexity.

In a different context, Paul [28] has investigated the so called inverse problem technique (also named the computed torque by Bejczy [29]). This approach uses the desired position, the desired velocity, and the desired acceleration to compute the driving torques. The main drawback of this scheme is that the computation of the complete nonlinear dynamic model is required. Simplifications have been obtained by Paul [30], Bejczy [31], Raibert and Horn[32] by omitting some of the terms in the model. These simplifications, while reducing the computational complexity, are still not enough for real time implementation of any control strategy based on this technique.

Vukobratovic and Stokic [33] have recognized that the forces (moments) acting on the robot joints can be directly measured and used to synthesize a feedback law that

compensates for the coupling in the robotics manipulator and relieves the controller from on line computation of these complex terms. Other attempts to include force feedback control account for the work of Hewit and Burdess [34] who introduced force transducers in the joints of the manipulator. Although the computation time is shorter, their scheme is still too complex for real time implementation. Wu and Paul [35] implemented an analog force feedback loop on a single joint manipulator which avoided the computational difficulty. Luh, Fisher, and Paul [36] have analyzed the effects of linear independent joint torques control. The stability of the overall system, however, has not been discussed in any of these papers. There were other attempts to use force feedback, not only at the executive control level but to include assembling tasks, such as in the resolved motion control introduced by Whitney [37]; the resolved acceleration control by Luh, Walker, and Paul [38]; and the resolved force control by Wu and Paul [39].

The simplest and most widely used control method today is based on decoupling and joint control. Yuan [40] tried to dynamically decouple a manipulator by linear control. An effective analysis of a constrained linear control may be found in Golla, Garg, and Hughes [41]. Freund [42] attempted the decoupling by nonlinear control involving full state feedback which guarantees stability in the absence of external disturbances. Young [43] developed a variable structure controller for manipulators.

b. Adaptive techniques

In addition to the computational complexity, the servomechanism approaches cannot always satisfy the stability conditions even if designed to be robust with respect to parametric and state disturbances. Adaptive control methodologies aim at overcoming these difficulties.

Within the adaptive control theory, two fundamental approaches exist in the literature [44]. The first is the Learning Model Adaptive Control (LMAC), in which an improved model of the plant is obtained by on line parameter estimation techniques, and is

then used in the feedback control. A general structure of this approach is shown in Figure 1.4. The estimated model and the controller may be either linear or nonlinear depending on the estimation technique used. The well known Self Tuning Regulator method belongs to this class. The second approach in adaptive control theory is the Model Reference Adaptive Control (MRAC). The controller is adjusted so that the dynamics of the closed loop system matches that of a preselected model. A general structure of this methodology is given in Figure 1.5. In general, the reference model is chosen to be a stable, linear, time-invariant, decoupled system. The controller may be either linear or nonlinear. It is also possible to design adaptive schemes which combine both techniques.

Many different structures of self tuning regulators are available in the literature, and they differ in parameter estimation technique and control algorithms [45]. Koivo and Guo [46] examined the feasibility of least squares techniques to robotics applications. Their approach is configuration dependent and does not account for nonlinearities in the system. It is based on estimation of the linearized dynamics and does not take advantage of any a priori knowledge of the system that might be available to the designer. Elliot, Depkovich, and Drapper [47] gave an extension of this method to the nonlinear case taking advantage of the fact that, in spite of their nonlinear nature, the parameters in the dynamic equations of a robot system appear linearly. This method showed better tracking ability, but did not solve the computation complexity. Cristi, Das, and Loh [48] exploited this idea of linear parameterization of the dynamic equations to give one of the first attempts to formulate an adaptive version of the computed torque technique. Their scheme consisted of an on line loop to estimate the payload and an off line loop to estimate the other parameters of the manipulator. The good feature of this approach is that it guarantees global stability of the system. In similar fashion, Craig, Hsu and Sastry [49] proposed another adaptive computed torque controller version, based on linear parameterization of the dynamic equations, and have established global convergence for their scheme. This method assumes no a priori

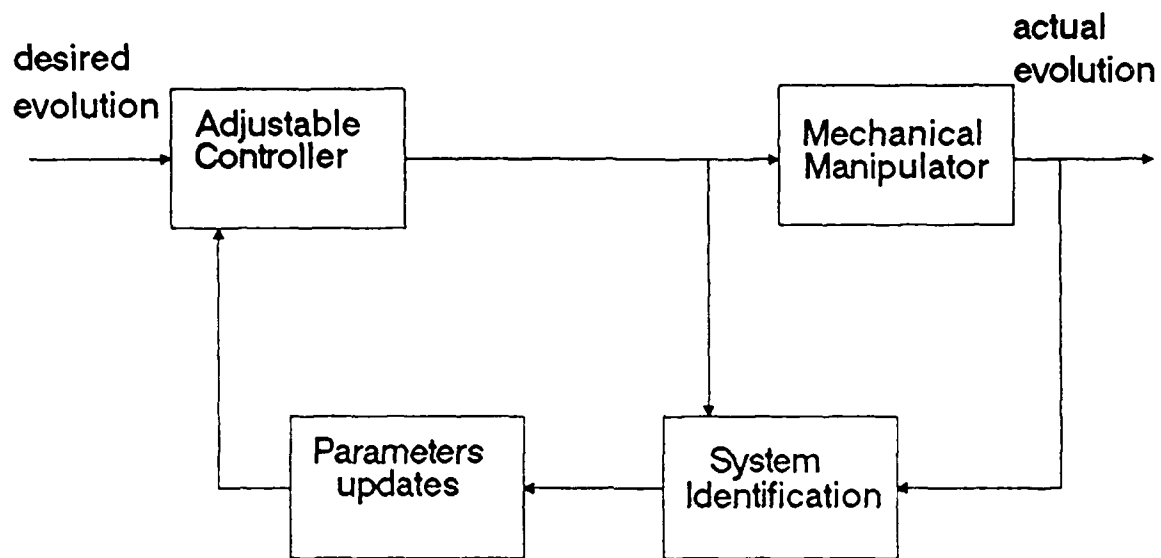


Figure 1.4: A Learning Model Adaptive Control (LMAC) structure.

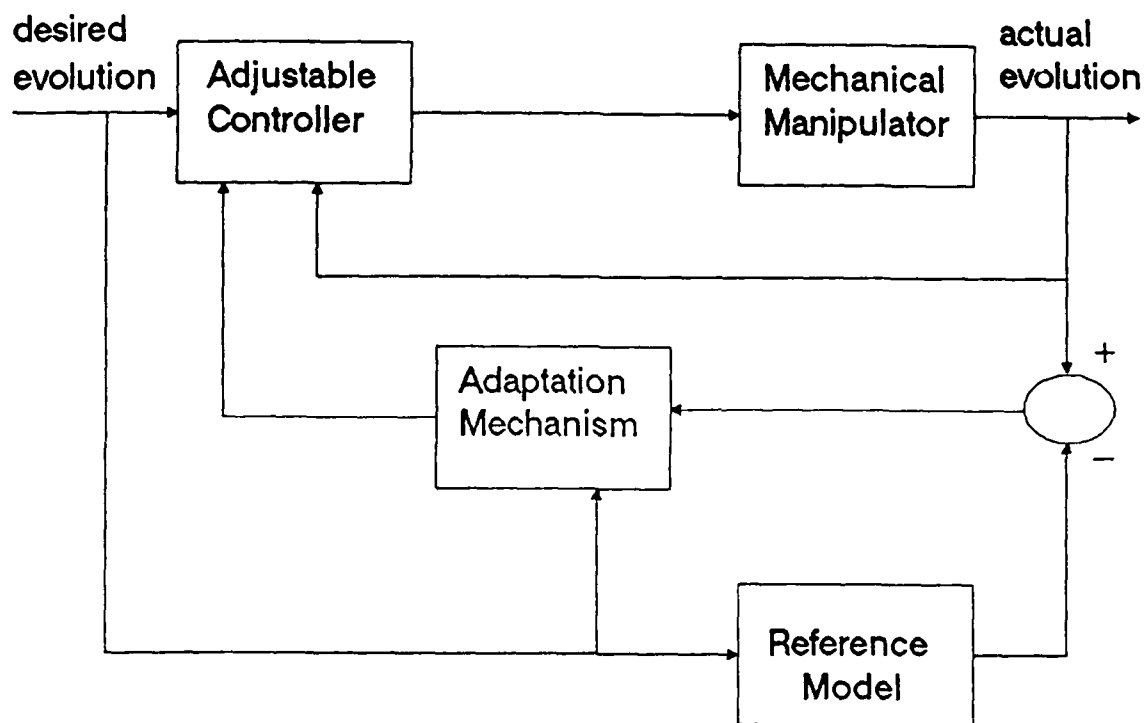


Figure 1.5: A Model Reference Adaptive Control (MRAC) structure.

knowledge about the system and requires acceleration measurements, which makes it numerically complex. Middleton and Goodwin [50] gave an extension of this method based on position and velocity measurements only. This method is still complex for real-time implementation. Lee and Chung [51] exploited the self-tuning regulator structure by introducing the adaptation at the level of linearized perturbation equations in the vicinity of a nominal joint trajectory. Their approach uses the recursive Newton-Euler equations for feedforward computation of nominal control and a recursive least squares, one step ahead control for feedback corrections about the nominal trajectory. The number of computations involved is reduced. However, this method can only compensate for small deviations. An attempt to speed up this method by avoiding the use of the Newton-Euler recursion was performed by Vukobratovic and Kircanski [52]. Their scheme is based on local adjustable controllers at each joint. The controller consists of a nominally tuned feedforward PID structure, and a feedback corrective portion to account for parameter variations. This method relaxes the computational burden by introducing a computer for each link, instead of one main computer for the whole robot system.

There are four basic approaches to the design of Model Reference Adaptive Control Systems [53]:

1. Local parametric optimization theory
2. Lyapunov functions
3. Variable structure systems
4. Hyperstability and positivity concepts

Within the local parametric optimization techniques, Dubowsky and DesForges [54] used the steepest descent method to develop one of the first contributions in adaptive control for robot manipulators. This method is computationally less burdensome and has good noise rejection properties. However, the steepest descent algorithm, while it can yield better adaptation speed, calls for many simplifications and may negatively influence the

overall stability of the manipulator. The input signal may also become excessively large due to the fact that only the output error is minimized. The discrete time version of this method [55], as well as the multivariable case [56], have also been developed by Dubowsky. The later approach was tested on an industrial robot and showed the significance of adaptive control in high speed tracking operations.

Takegaki and Arimoto [57] have considered the applicability of model reference adaptive control theory in robotics, using the Lyapunov function approach. Their scheme included a nonadaptive gravity compensation loop. This resulted in simple adaptation and control laws, thus making it suitable for real time implementation. However, how the gravity compensation loop affected the tracking quality could not be shown.

Young [58] combined the variable structure theory with the model-following approach and investigated their use in robot positioning problems. This approach also uses less computation. It, however, uses the hierarchical control methodology of Utkin [59], which may not be valid if only asymptotic convergence can be reached. In addition, the control signals are discontinuous and the high frequency components may become unacceptable. This method suffers also from the fact that there are no design procedures for tuning the controller parameters. Slotine and Sastry tried [60] to remove some of these difficulties by using the concept of time varying sliding surfaces in the state space. They, however, trade off accuracy against chattering by approximating the obtained discontinuous control law by a continuous one.

Horowitz and Tomizuka [61] presented one of the first attempts to apply hyperstability theory to robotics. Their algorithm has been later implemented on a three degree of freedom manipulator by Anex and Hubbard [62] after been slightly modified to compensate for gravity. The advantage of this method is that the adaptation mechanism is derived from the condition of overall system stability. Its main problem is the fact that the dynamical effects are estimated without using any a priori knowledge about the system

dynamics. In practice, many of the robot parameters are known and it would be convenient to estimate only the unknown ones. The application of hyperstability theory to robotics models has been fully developed by Balestrino, De Maria, and Sciavicco [63]. Their strategy offers better transient behavior compared to that with self-tuning regulators and it guarantees stability of the entire system. The main drawback of this method is the possibility of excessive control signals and its high numerical complexity.

As a conclusion, let us note that adaptive methods for manipulation robot are still in their early stages of development. It is, therefore, very difficult to produce an exhaustive survey of these methods as new design ideas continue to appear in the literature. So far we have summarized some of the approaches available, far from a complete treatment of the problem.

G. THESIS OUTLINE

The remaining of this research will be centered around the development of adaptive control strategies applied to robotics systems. Fundamental to the problem of dynamic control, the derivation of the dynamic equations of motion will be addressed in Chapter Two. Different approaches to obtaining these equations, as well as their computational complexity will be discussed.

In Chapter Three, a new adaptive control law which will combine properties from both the STR technique in [51] and the hyperstability principal in [63] will be presented. This methodology has the advantage of assuring global stability and aims at overcoming many of the limitations of the previously studied schemes. It makes use of a nominal dynamics feedforward compensation loop, but, unlike Ref.[51], the stabilizing feedback loop does not call for any simplifying assumptions. A rapprochement between this method and the AMFC will be established, but, unlike Ref.[63], it does not require excessive actuation. Most of all, our approach yields better performance and is numerically very efficient.

II. MECHANICAL MANIPULATOR DYNAMICS

A. INTRODUCTION

In order to design a controller of an articulated mechanical system, it is necessary to have a mathematical model the system. This model expresses the relationships among different components of the robotics system and the interactions between the mechanical manipulator as a whole and the physical universe surrounding it. It is described in terms of characteristic variables which are specific to the system, such as degrees of freedom, lengths, masses, inertias, positions, forces, and torques.

The number and nature of the parameters used in each model depend on the application and the accuracy required. The designer is constantly faced with the challenge of developing models that adequately represent the dynamics of the system, and that are computationally convenient for computer implementations.

Because of the high speeds required in any future robotics application, dynamic phenomena, such as frictional, inertial, centrifugal, and coupling forces should be taken into consideration for the chosen model to be representative of the actual mechanical manipulator behavior.

An efficient mathematical model of a robot is essential for both design and control purposes. In the design phase, a complete dynamic model is useful for determining loads, dimensions, tolerances and actuation. In control applications, the dynamic model is used to generate the nominal joint torques as well as to simulate and test control strategies without the need of building a prototype (at least in the early stages of the design).

There are two problems related to the dynamics of a manipulator. In the inverse dynamics problem, we are given a trajectory in terms of joint coordinates $\mathbf{q}(t)$ and their derivatives, $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$, and we wish to find the corresponding sets of vector torques $\boldsymbol{\tau}$.

This formulation is at the basis of the control problem. The second formulation is the direct dynamics problem. In this formulation, we wish to calculate the resulting motion of the manipulator $q(t)$, $\dot{q}(t)$ and $\ddot{q}(t)$ for every given set of vector torques τ . This is at the basis of the simulation of the robotics system.

There are several possible approaches one can take to derive the dynamics equations of an articulated mechanical system. Newton-Euler's equations, Gibbs' functions, d'Alembert's formalism, Bond graphs and Lagrange equations are only few of these methods.

Lagrange and Newton-Euler equations are, however, the most frequently used in the literature. In this chapter, we will present several alternative formulations of these two methods, address their computational performances, and show that one can easily be derived from the other.

B. CLOSED-FORM LAGRANGIAN MANIPULATOR DYNAMICS

This formulation was first applied to open loop kinematic chains by Kahn [64] from the more general linkage problem of Uicker [65], and has served as the standard manipulator dynamics for over a decade. We begin this derivation by presenting the notation used throughout the development.

The links of a manipulator are numbered consecutively from 1 to n starting from the base to the tip. By convention, the reference frame is numbered as link 0. The joints are numbered so that the joint i connects link $i-1$ to link i . An orthogonal coordinate system is fixed in each link as follows:

- z_i is directed along the axis of joint $i+1$,
- x_i lies along the common normal from z_{i-1} to z_i , and
- y_i completes the right handed coordinate.

The relative position of two adjacent links is completely described by:

- a_i , the distance between the origins of coordinate systems $i-1$ and i measured along x_i ,
- s_i , the distance between x_{i-1} and x_i measured along z_{i-1} ,
- α_i , the angle between the z_{i-1} and z_i axes measured in a righthand sense about x_i , and
- θ_i , the angle between the x_{i-1} and x_i axes measured in the righthand sense about z_{i-1} .

This notation is summarized in Figure 2.1. If the joint is rotational, the joint variable will be θ_i ; if translational, the joint variable will be s_i . The symbol q_i will designate the

variable for joint i whether it is s_i or θ_i . The vector $\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}$ represents the generalized

coordinates of the manipulator and completely specifies its position. In the subsequent development, lower case and uppercase regular letters will be used indifferently to designate scalar quantities, lower case bold letters to designate vectors, and capital bold letters to designate matrices. Subscripts refer to the physical location of the variable, superscripts to the coordinates frame the variable is expressed in. Either of these is omitted when referring to the base coordinates frame.

The Lagrange equations for a nonconservative system are:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i, \quad i=1,2,\dots,n \quad (2.1)$$

where.

$L = K - P$ is the Lagrangian function,

K is the total kinetic energy of the manipulator,

P is the total potential energy of the manipulator,

q_i is the generalized coordinate of the manipulator,

\dot{q}_i is the first time derivative of the generalized coordinate, and,

τ_i is the torque applied to the system at joint i .

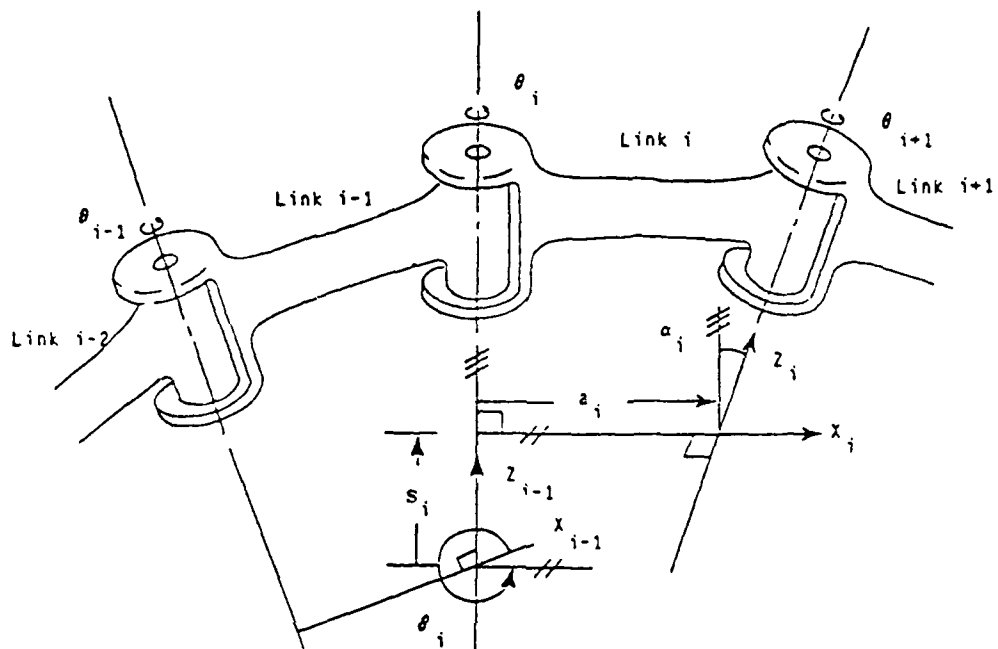


Figure 2.1: The standard axes definitions for connected links

To find the kinetic energy of the physical system, we need to know the velocity of each

joint. Let $\mathbf{p}_i^i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$ denote the coordinates of link i in the reference frame of the i^{th} link;

\mathbf{p}_i , the same point \mathbf{p}_i^i with respect to the base coordinates frame; $\mathbf{T}_i^{i-1} = \begin{bmatrix} \mathbf{R}_i^{i-1} & \lambda_i^{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$, the homogeneous coordinate transformation which relates the displacement of the i^{th} link coordinate frame to the $(i-1)^{th}$ link coordinate frame; and \mathbf{T}_i , the coordinate transformation which relates the i^{th} coordinate frame to the base coordinate frame. The rotational transformation \mathbf{R}_i^{i-1} and the translational transformation λ_i^{i-1} are given by:

$$\mathbf{R}_i^{i-1} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cos \alpha_i & \sin \theta_i \sin \alpha_i \\ \sin \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \sin \alpha_i \\ 0 & \sin \alpha_i & \cos \alpha_i \end{bmatrix} \quad (2.2)$$

and,

$$\lambda_i^{i-1} = \begin{bmatrix} \rho a_i \cos \theta_i \\ \rho a_i \sin \theta_i \\ s_i \end{bmatrix}, \rho = \begin{cases} 1 & \text{for a revolute joint} \\ 0 & \text{for a prismatic joint} \end{cases} \quad (2.3)$$

The variables \mathbf{p}_i and \mathbf{p}_i^i are related by:

$$\mathbf{p}_i = \mathbf{T}_i \mathbf{p}_i^i \quad (2.4)$$

where

$$\mathbf{T}_i = \mathbf{T}_1^0 \mathbf{T}_2^1 \dots \mathbf{T}_i^{i-1} \quad (2.5)$$

Assuming rigid body motion, all the points \mathbf{p}_i^i will have zero velocity with respect to the i^{th} coordinate frame. The velocity of \mathbf{p}_i^i expressed in the base coordinate frame is:

$$\mathbf{v}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{z}_i \\ 0 \end{bmatrix} = \frac{d}{dt}(\mathbf{p}_i) = \frac{d}{dt}(\mathbf{T}_i \mathbf{p}_i^i) \quad (2.6)$$

Since $\dot{\mathbf{p}}_i^i = 0$, \mathbf{v}_i can be written as:

$$\mathbf{v}_i = \left[\sum_{j=1}^i \frac{\partial \mathbf{T}_i}{\partial q_j} \dot{q}_j \right] \mathbf{p}_i^i \quad (2.7)$$

where

$$\frac{\partial \mathbf{T}_i^{i-1}}{\partial q_i} = \mathbf{Q}_i \mathbf{T}_i^{i-1}, \quad (2.8)$$

and

$$\frac{\partial \mathbf{T}_i}{\partial q_j} = \begin{cases} \mathbf{T}_1^0 \mathbf{T}_2^1 \dots \mathbf{T}_{j-1}^{j-2} \mathbf{Q}_j \mathbf{T}_j^{j-1} \dots \mathbf{T}_i^{i-1}, & \text{for } j \leq i \\ 0 & \text{for } j > i \end{cases} \quad (2.9)$$

$i=1, 2, \dots, n$

The matrix \mathbf{Q}_i being:

$$\mathbf{Q}_i = \begin{bmatrix} 0 & -\nu & 0 & 0 \\ \nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu \\ 0 & 0 & 0 & 0 \end{bmatrix}, \text{ with } \begin{cases} \nu=1 \text{ and } \mu=0 \text{ for} \\ \text{a revolute joint} \\ \nu=0 \text{ and } \mu=1 \text{ for} \\ \text{a prismatic joint} \end{cases} \quad (2.10)$$

In order to simplify notations, let us define $\mathbf{U}_{ij} = \frac{\partial \mathbf{T}_i}{\partial q_j}$, then equation (2.9) can be

written as follows:

$$\mathbf{U}_{ij} = \begin{cases} \mathbf{T}_{j-1} \mathbf{Q}_j \mathbf{T}_i^{j-1}, & \text{for } j \leq i \\ 0 & \text{for } j > i \end{cases}, \quad i=1, 2, \dots, n \quad (2.11)$$

Using this notation, \mathbf{v}_i can be expressed as:

$$\mathbf{v}_i = \left[\sum_{j=1}^i \mathbf{U}_{ij} \dot{q}_j \right] \mathbf{p}_i^i \quad (2.12)$$

The matrix \mathbf{U}_{ij} is the rate of change of the point \mathbf{p}_i^i on link i relative to the base coordinate frame as q_j changes. It represents both the linear and angular velocities of the link.

The kinetic energy of an infinitesimal mass dm on link i is found as:

$$\begin{aligned} dk_i &= \frac{1}{2} (\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2) dm \\ dk_i &= \frac{1}{2} \text{Tr}(\mathbf{v}_i \mathbf{v}_i^T) dm \end{aligned} \quad (2.13)$$

where a trace of an $n \times n$ matrix \mathbf{A} is defined as:

$$\text{Tr}(\mathbf{A}) = \sum_{i=1}^n a_{ii}$$

Substituting equation (2.12) for \mathbf{v}_i , the kinetic energy of the infinitesimal mass becomes:

$$dk_i = \frac{1}{2} \text{Tr} \left[\sum_{\lambda=1}^i \sum_{k=1}^i U_{i\lambda} (\mathbf{p}_i^i dm \mathbf{p}_i^{iT}) U_{ik}^T \dot{q}_\lambda \dot{q}_k \right] \quad (2.14)$$

For the whole link,

$$\begin{aligned} K_i &= \int dk_i \\ K_i &= \frac{1}{2} \text{Tr} \left[\sum_{\lambda=1}^i \sum_{k=1}^i U_{i\lambda} \left(\int \mathbf{p}_i^i \mathbf{p}_i^{iT} dm \right) U_{ik}^T \dot{q}_\lambda \dot{q}_k \right] \end{aligned} \quad (2.15)$$

The integral term inside the bracket is known as the pseudo inertia matrix \mathbf{J}_i^i of all the points on link i with respect to the proximate joint of link i expressed in the i^{th} link coordinates system.

$$\begin{aligned} \mathbf{J}_i^i &= \int \mathbf{p}_i^i \mathbf{p}_i^{iT} dm \\ \mathbf{J}_i^i &= \begin{bmatrix} \int x_i^2 dm & \int x_i y_i dm & \int x_i z_i dm & \int x_i dm \\ \int x_i y_i dm & \int y_i^2 dm & \int y_i z_i dm & \int y_i dm \\ \int x_i z_i dm & \int y_i z_i dm & \int z_i^2 dm & \int z_i dm \\ \int x_i dm & \int y_i dm & \int z_i dm & \int dm \end{bmatrix} \end{aligned} \quad (2.16)$$

The inertia tensor \mathbf{I}_{uv}^i of link i about its center of mass in the i^{th} coordinate frame is defined as:

$$\mathbf{I}_{uv}^i = \int \left[\delta_{uv} \left[\sum_k x_k^2 \right] - x_u x_v \right] dm$$

where the indices u, v, k indicate principal axes of the i^{th} coordinate frame and δ_{uv} is the

2-D Kronecker delta. The pseudo inertia matrix J_i^i can then be expressed in terms of the inertia tensor I_{uv}^i as:

$$J_i^i = \begin{bmatrix} \frac{-I_{xx}^i + I_{yy}^i + I_{zz}^i}{2} & I_{xy}^i & I_{xz}^i & m_i x_{c_i} \\ I_{xy}^i & \frac{I_{xx}^i - I_{yy}^i + I_{zz}^i}{2} & I_{yz}^i & m_i y_{c_i} \\ I_{xz}^i & I_{yz}^i & \frac{I_{xx}^i + I_{yy}^i - I_{zz}^i}{2} & m_i z_{c_i} \\ m_i x_{c_i} & m_i y_{c_i} & m_i z_{c_i} & m_i \end{bmatrix} \quad (2.17)$$

where $c_i^i = \begin{bmatrix} x_{c_i} \\ y_{c_i} \\ z_{c_i} \\ 1 \end{bmatrix}$ is the center of mass vector of link i from the i^{th} link coordinate frame and expressed in the i^{th} link coordinates system.

The total kinetic energy K of the manipulator arm can be expressed as:

$$K = \sum_{i=1}^n K_i$$

$$K = \frac{1}{2} \sum_{i=1}^n \sum_{\lambda=1}^i \sum_{k=1}^i \left[\text{Tr}(\mathbf{U}_{i\lambda} \mathbf{J}_i^i \mathbf{U}_{ik}^T) \dot{q}_\lambda \dot{q}_k \right] \quad (2.18)$$

Note that the terms J_i^i are dependent on the mass distribution of link i and not on their position or rate of motion. Hence, the J_i^i need to be computed only once for evaluating the kinetic energy of the manipulator.

The potential energy P_i of the link i is:

$$P_i = -m_i \mathbf{g} \mathbf{p}_i = -m_i \mathbf{g} \left[\mathbf{T}_i \mathbf{p}_i^i \right] \quad (2.19)$$

where $\mathbf{g} = [g_x \ g_y \ g_z \ 0]$ is the gravity row vector expressed in the base coordinate system.

The total potential energy of the manipulator then becomes:

$$P = \sum_{i=1}^n P_i = - \sum_{i=1}^n m_i g [T_i p_i^i] \quad (2.20)$$

and the Lagrangian function L then becomes:

$$L = \frac{1}{2} \sum_{i=1}^n \sum_{\lambda=1}^i \sum_{k=1}^i \left[\text{Tr}(U_{i\lambda} J_i^i U_{ik}^T) \dot{q}_\lambda \dot{q}_k \right] + \sum_{i=1}^n m_i g [T_i p_i^i] \quad (2.21)$$

Performing the differentiation in the Lagrange equations and rearranging, we obtain the necessary generalized torque τ_i for joint i actuator to drive the i^{th} link of the manipulator.

$$\tau_i = \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i}$$

$$\tau_i = \sum_{\lambda=ik=1}^n \sum_{\lambda}^{\lambda} \text{Tr}(U_{\lambda k} J_{\lambda}^{\lambda} U_{\lambda i}^T) \ddot{q}_k + \sum_{\lambda=ik=1}^n \sum_{\lambda}^{\lambda} \sum_{l=1}^{\lambda} \text{Tr}(U_{\lambda k l} J_{\lambda}^{\lambda} U_{\lambda i}^T) \dot{q}_\lambda \dot{q}_l - \sum_{\lambda=i}^n m_{\lambda} g U_{\lambda i} p_{\lambda}^{\lambda}$$

$$i = 1, 2, \dots, n \quad (2.22)$$

The above equation can be expressed in a matrix notation as:

$$\tau_i = \sum_{k=1}^n a_{ik} \ddot{q}_k + \sum_{k=1}^n \sum_{l=1}^n v_{ikl} \dot{q}_k \dot{q}_l + g_i, \quad i=1,2,\dots,n \quad (2.23)$$

or in more compact form as:

$$\tau(t) = A[q(t)] \ddot{q}(t) + V[q(t), \dot{q}(t)] + G[q(t)] \quad (2.24)$$

where

$$\tau(t) = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_n \end{bmatrix} \text{ is an } n \times 1 \text{ generalized torque vector applied at joints } i=1,2,\dots,n,$$

$$q(t) = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \text{ is an } n \times 1 \text{ vector of the joint variables of the manipulator,}$$

$\dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix}$ is an $n \times 1$ vector of the joint velocity of the manipulator,

$\ddot{\mathbf{q}}(t) = \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \vdots \\ \ddot{q}_n \end{bmatrix}$ is an $n \times 1$ vector of the acceleration of the joint variables, and

$\mathbf{A}[\mathbf{q}(t)]$ is an $n \times n$ inertial acceleration related symmetric matrix whose elements are:

$$a_{ik} = \sum_{\lambda=\max(i,k)}^n \text{Tr}(\mathbf{U}_{\lambda k} \mathbf{J}_{\lambda}^{\lambda} \mathbf{U}_{\lambda i}^T), \quad i, k=1, 2, \dots, n \quad (2.24)$$

When $i=k$, a_{ii} is related to the acceleration of joint i where τ_i acts and is known as effective inertia. When $i \neq k$, a_{ik} is related to the reaction torque induced by the acceleration of joint k and acting at joint i (known as coupling inertia), and

$\mathbf{V}[\mathbf{q}(t), \dot{\mathbf{q}}(t)] = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$ is an $n \times 1$ velocity related vector composed of Coriolis and centrifugal forces, where

$$v_i = \sum_{k=1}^n \sum_{l=1}^n v_{ikl} \dot{q}_k \dot{q}_l, \quad i=1, 2, \dots, n \quad (2.25)$$

and

$$v_{ikl} = \sum_{\lambda=\max(i,k)}^n \text{Tr}(\mathbf{U}_{\lambda k} \mathbf{J}_{\lambda}^{\lambda} \mathbf{U}_{\lambda i}^T), \quad i, k, l=1, 2, \dots, n \quad (2.26)$$

When $k=l$, the velocity torques are known as centripetal torques, and when $k \neq l$ as Coriolis torques. Friction torques are also velocity related and can be added to this term as:

$$\text{fr}_i = C \text{sgn}(\dot{q}_i) + \mathbf{V} \dot{\mathbf{q}}_i$$

where

C is a coulomb friction constant,

V is a viscous friction constant and,

sgn is the sign function, and

$G[q(t)] = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix}$ is an $n \times 1$ gravity loading force vector, where,

$$g_i = - \sum_{\lambda=i}^n m_{\lambda} g U_{\lambda i} p_{\lambda}^{\lambda} \quad (2.27)$$

The dynamic equations of motion as given by equation (2.24) are coupled, nonlinear, second order, ordinary differential equations. Notice also that equation (2.24) yields the solution of the inverse dynamics problem. For every point $(q(t), \dot{q}(t), \ddot{q}(t))$ of a given trajectory, it yields the required joint torques vector τ . This form allows design of a control law that easily compensates for the nonlinear effects. Computationally, however, these equations are extremely inefficient as they require:

1. $(\frac{128}{3})n^4 + (\frac{512}{3})n^3 + (\frac{844}{3})n^2 + (\frac{76}{3})n$ multiplications, and,
2. $(\frac{98}{3})n^4 + (\frac{781}{3})n^3 + (\frac{637}{3})n^2 + (\frac{107}{3})n$ additions.

for every set point in the trajectory. That is, they are of $O(n^4)$ order of complexity, where n is the number of links.

There are two categories of approaches in trying to implement the closed-form Lagrange equations in real time control applications:

1. Simplifying the dynamics by ignoring the least significant terms and correcting errors with some feedback compensation. The simplifying assumptions, however, may not hold for all speeds and all ranges of applications.
2. Precomputing terms in the equations and using a gain scheduling approach.

C. RECURSIVE LAGRANGIAN MANIPULATOR DYNAMICS

The main reason for the inefficiency of the Uicker/Khan formulation is due to the fact that these equations are closed-form expressions and most of the terms are reevaluated

many times. To reduce the complexity of these equations, we need to reexamine the above derivation and recast it into a recursive form which is computationally more efficient [66].

The generalized driving torques given in equation (2.22) can be expressed as:

$$\tau_i = \sum_{\lambda=i}^n \left[\text{Tr}(U_{\lambda i} J_{\lambda}^{\lambda} \ddot{T}_{\lambda}^T) - m_{\lambda} g U_{\lambda i} p_{\lambda}^{\lambda} \right], \quad i=1,2,\dots,n \quad (2.28)$$

where \ddot{T}_{λ} is defined as:

$$\ddot{T}_{\lambda} = \sum_{k=1}^{\lambda} U_{\lambda k} \ddot{q}_k + \sum_{k=1}^{\lambda} \sum_{l=1}^{\lambda} U_{\lambda k l} \dot{q}_k \dot{q}_l \quad (2.29)$$

The advantage of the above substitution is that equation (2.29) is never used in the computation. More efficient recurrence relations for the velocity \dot{T}_{λ} and acceleration \ddot{T}_{λ} are easily derived by straightforward differentiation:

$$\dot{T}_{\lambda} = \dot{T}_{\lambda-1} T_{\lambda}^{\lambda-1} \quad (2.30)$$

$$\begin{aligned} \dot{T}_{\lambda} &= \dot{T}_{\lambda-1} T_{\lambda}^{\lambda-1} + T_{\lambda-1} \dot{T}_{\lambda}^{\lambda-1} \\ \dot{T}_{\lambda} &= \dot{T}_{\lambda-1} T_{\lambda}^{\lambda-1} + T_{\lambda-1} U_{\lambda \lambda} \dot{q}_{\lambda} \end{aligned} \quad (2.31)$$

$$\begin{aligned} \ddot{T}_{\lambda} &= \ddot{T}_{\lambda-1} T_{\lambda}^{\lambda-1} + \dot{T}_{\lambda-1} \dot{T}_{\lambda}^{\lambda-1} + \dot{T}_{\lambda-1} U_{\lambda \lambda} \dot{q}_{\lambda} + T_{\lambda-1} \ddot{U}_{\lambda \lambda} \dot{q}_{\lambda} + T_{\lambda-1} U_{\lambda \lambda} \ddot{q}_{\lambda} \\ \ddot{T}_{\lambda} &= \ddot{T}_{\lambda-1} T_{\lambda}^{\lambda-1} + 2\dot{T}_{\lambda-1} U_{\lambda \lambda} \dot{q}_{\lambda} + T_{\lambda-1} U_{\lambda \lambda \lambda} \dot{q}_{\lambda}^2 + T_{\lambda-1} U_{\lambda \lambda} \ddot{q}_{\lambda} \end{aligned} \quad (2.32)$$

Computing the driving torques as given by equation (2.28), and using the recursive relations in (2.30), (2.31), and (2.32), results in an $O(n^2)$ order of complexity, requiring:

1. $106 \frac{1}{2} n^2 + 620 \frac{1}{2} n - 512$ multiplications, and,
2. $82 n^2 + 514 n - 384$ additions.

The reduction in complexity comes from the fact that to calculate coriolis and centrifugal forces, we only need to calculate $U_{\lambda \lambda \lambda}$ instead of all the matrices $U_{\lambda k l}$

Further computations can be saved by noting that:

$$U_{\lambda i} = U_{ii} T_{\lambda}^i \quad (2.33)$$

Therefore, the generalized torques equation (2.28) can be written:

$$\begin{aligned}\tau_i &= \sum_{\lambda=i}^n \left[\text{Tr}(\mathbf{U}_{ii} \mathbf{T}_{\lambda}^i \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda}^T) - m_{\lambda} g \mathbf{U}_{ii} \mathbf{T}_{\lambda}^i \mathbf{p}_{\lambda}^{\lambda} \right] \\ \tau_i &= \text{Tr}(\mathbf{U}_{ii} \sum_{\lambda=i}^n \mathbf{T}_{\lambda}^i \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda}^T) - g \mathbf{U}_{ii} \sum_{\lambda=i}^n m_{\lambda} \mathbf{T}_{\lambda}^i \mathbf{p}_{\lambda}^{\lambda}\end{aligned}\quad (2.34)$$

or

$$\tau_i = \text{Tr}(\mathbf{U}_{ii} \mathbf{D}_i) - g \mathbf{U}_{ii} \mathbf{C}_i \quad (2.35)$$

where

$$\begin{aligned}\mathbf{D}_i &= \sum_{\lambda=i}^n \mathbf{T}_{\lambda}^i \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda}^T \\ \mathbf{D}_i &= \mathbf{T}_i^i \mathbf{J}_i^i \ddot{\mathbf{T}}_i^T + \sum_{\lambda=i+1}^n \mathbf{T}_{i+1}^i \mathbf{T}_{\lambda}^{i+1} \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda}^T \\ \mathbf{D}_i &= \mathbf{J}_i^i \ddot{\mathbf{T}}_i^T + \mathbf{T}_{i+1}^i \mathbf{D}_{i+1}\end{aligned}\quad (2.36)$$

and

$$\begin{aligned}\mathbf{C}_i &= \sum_{\lambda=i}^n m_{\lambda} \mathbf{T}_{\lambda}^i \mathbf{p}_{\lambda}^{\lambda} \\ \mathbf{C}_i &= m_i \mathbf{p}_i^i + \mathbf{T}_{i+1}^i \mathbf{C}_{i+1}\end{aligned}\quad (2.37)$$

These recursive relations can be computed as follows:

For $i = 1, 2, \dots, n$

1. compute \mathbf{T}_i by equation (2.30)
2. compute $\dot{\mathbf{T}}_i$ by equation (2.31)
3. compute $\ddot{\mathbf{T}}_i$ by equation (2.32)
4. if $i = n$, continue. Otherwise, set $i=i+1$ and return to 1.
5. compute \mathbf{D}_i by equation (2.36)
6. compute \mathbf{C}_i by equation (2.37)
7. compute τ_i by equation (2.35)
8. if $i = 1$, stop. Otherwise, set $i=i-1$, and return to 5.

The advantage of this formulation is that its complexity is now $O(n)$. It requires:

1. $830n - 592$ multiplications, and,
2. $675n - 464$ additions.

Any other reduction in computational complexity can only be obtained by reducing the size of the coefficients in the above complexity polynomials [67]. This can be achieved through reformulating the Lagrangian dynamics in terms of 3x3 rotational matrices rather than 4x4 rotation-translation matrices. Because 3x3 matrix multiplications require 27 multiplications while 4x4 matrix multiplications require 64, we get a greater than 50% reduction in the coefficients of the computational cost terms.

The matrix T_i^{i-1} relating the orientation of the coordinate system $i-1$ and i is now reduced to a 3x3 rotation matrix R_i^{i-1} . A point on link i , expressed in homogeneous coordinates with respect to the i^{th} coordinates frame, is now represented as $p_i^i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$, with p_i the same point with respect to the base coordinate frame. The following sets of vectors are also needed throughout this derivation :

- o_i^j , the joint i coordinate origin expressed in the j^{th} coordinates frame, and,
- c_i^j , the link i center of mass expressed in the j^{th} coordinates frame.

The quantities, p_i , o_i and p_i^i are related by:

$$p_i = o_i + T_i p_i^i \quad (2.38)$$

The velocity of p_i is then given by:

$$\dot{p}_i = v_i = \dot{o}_i + \dot{T}_i p_i^i \quad (2.39)$$

and the kinetic energy for a particle on link i is as given earlier:

$$dk_i = \frac{1}{2} \text{Tr}(v_i v_i^T) dm \quad (2.40)$$

$$dk_i = \frac{1}{2} \text{Tr}(\dot{o}_i \dot{o}_i^T + 2\dot{T}_i p_i^i \dot{o}_i^T T_i p_i^i \dot{T}_i^T p_i^i \dot{T}_i^T p_i^i) dm \quad (2.41)$$

Integrating over all particles in link i , the total kinetic energy K_i of link i is given by:

$$K_i = \frac{1}{2} \text{Tr}(m_i \dot{\mathbf{o}}_i \dot{\mathbf{o}}_i^T + 2 \dot{\mathbf{T}}_i \mathbf{n}_i^i \dot{\mathbf{o}}_i^T + \dot{\mathbf{T}}_i \mathbf{J}_i^i \dot{\mathbf{T}}_i^T) \quad (2.42)$$

where

$$\mathbf{n}_i^i = \int \mathbf{p}_i^i dm \quad (2.43)$$

Therefore, the total kinetic energy for all the links is given by:

$$K = \frac{1}{2} \sum_{\lambda=1}^n \text{Tr}(m_\lambda \dot{\mathbf{o}}_\lambda \dot{\mathbf{o}}_\lambda^T + 2 \dot{\mathbf{T}}_\lambda \mathbf{n}_\lambda^\lambda \dot{\mathbf{o}}_\lambda^T + \dot{\mathbf{T}}_\lambda \mathbf{J}_\lambda^\lambda \dot{\mathbf{T}}_\lambda^T) \quad (2.44)$$

and the potential energy becomes:

$$P = \sum_{\lambda=1}^n m_\lambda g \mathbf{T}_\lambda \mathbf{p}_\lambda^\lambda \quad (2.45)$$

Proceeding as in the homogeneous coordinate formulation, we can write the Lagrange equations for each link as:

$$\tau_i = \frac{d}{dt} \frac{\partial K}{\partial \dot{\mathbf{q}}_i} - \frac{\partial K}{\partial \mathbf{q}_i} + \frac{\partial P}{\partial \mathbf{q}_i} \quad (2.46)$$

which take into account the fact that the potential energy depends on position only.

Performing the differentiation and rearranging terms yields:

$$\tau_i = \sum_{\lambda=i}^n \left[\text{Tr} \left(m_\lambda \frac{\partial \mathbf{o}_\lambda}{\partial \mathbf{q}_i} \ddot{\mathbf{o}}_\lambda^T + \frac{\partial \mathbf{o}_\lambda}{\partial \mathbf{q}_i} \mathbf{n}_\lambda^\lambda \ddot{\mathbf{T}}_\lambda^T + U_{\lambda i} \mathbf{n}_\lambda^\lambda \ddot{\mathbf{o}}_\lambda^T + U_{\lambda i} \mathbf{J}_\lambda^\lambda \ddot{\mathbf{T}}_\lambda^T - m_\lambda g \mathbf{T}_\lambda \mathbf{p}_\lambda^\lambda \right) \right] \quad (2.47)$$

Recall that

$$\mathbf{o}_\lambda = \mathbf{o}_i + \mathbf{T}_i \mathbf{o}_\lambda^i \quad (2.48)$$

$$\frac{\partial \mathbf{o}_\lambda}{\partial \mathbf{q}_i} = U_{ii} \mathbf{o}_\lambda^i \quad (2.49)$$

$$U_{\lambda i} = U_{ii} \mathbf{T}_\lambda^i \quad (2.50)$$

Substituting the above relations into equation (2.47), we obtain:

$$\tau_i = \text{Tr} \left(U_{ii} \sum_{\lambda=i}^n (m_\lambda \mathbf{o}_\lambda^i \ddot{\mathbf{o}}_\lambda^T + \mathbf{o}_\lambda^i \mathbf{n}_\lambda^\lambda \ddot{\mathbf{T}}_\lambda^T + \mathbf{T}_\lambda^i \mathbf{n}_\lambda^\lambda \ddot{\mathbf{o}}_\lambda^T + \mathbf{T}_\lambda^i \mathbf{J}_\lambda^\lambda \ddot{\mathbf{T}}_\lambda^T) - g U_{ii} \sum_{\lambda=i}^n m_\lambda \mathbf{T}_\lambda^i \mathbf{p}_\lambda^\lambda \right) \quad (2.51)$$

or

$$\tau_i = \text{Tr}(\mathbf{U}_{ii}\mathbf{D}_i) - g\mathbf{U}_{ii}\mathbf{C}_i \quad (2.52)$$

where

$$\mathbf{C}_i = \sum_{\lambda=i}^n m_{\lambda} \mathbf{T}_{\lambda}^i \mathbf{p}_{\lambda}^{\lambda} = m_i \mathbf{p}_i^i + \mathbf{T}_{i+1}^i \mathbf{C}_{i+1} \quad (2.53)$$

has the same structure as equation (2.37), except for the difference in dimensionality. For the term \mathbf{D}_i , we can write the recurrence:

$$\begin{aligned} \mathbf{D}_i &= \sum_{\lambda=i}^n (m_{\lambda} \mathbf{o}_{\lambda}^i \ddot{\mathbf{o}}_{\lambda} + \mathbf{o}_{\lambda}^i \mathbf{n}_{\lambda}^{\lambda} \mathbf{r} \ddot{\mathbf{T}}_{\lambda} + \mathbf{T}_{\lambda}^i \mathbf{n}_{\lambda}^{\lambda} \ddot{\mathbf{o}}_{\lambda} + \mathbf{T}_{\lambda}^i \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda}) \\ \mathbf{D}_i &= \sum_{\lambda=i+1}^n ((\mathbf{T}_{i+1}^i \mathbf{o}_{\lambda}^{i+1} + \mathbf{o}_{i+1}^i)(m_{\lambda} \ddot{\mathbf{o}}_{\lambda} + \mathbf{n}_{\lambda}^{\lambda} \mathbf{r} \ddot{\mathbf{T}}_{\lambda}) + \mathbf{T}_{i+1}^i \mathbf{T}_{\lambda}^{i+1} (\mathbf{n}_{\lambda}^{\lambda} \ddot{\mathbf{o}}_{\lambda} + \mathbf{J}_{\lambda}^{\lambda} \ddot{\mathbf{T}}_{\lambda})) + \mathbf{n}_i^i \ddot{\mathbf{o}}_i + \mathbf{J}_i^i \ddot{\mathbf{T}}_i \\ \mathbf{D}_i &= \mathbf{T}_{i+1}^i \mathbf{D}_{i+1} + \mathbf{o}_{i+1}^i \mathbf{e}_{i+1} \mathbf{n}_i^i \ddot{\mathbf{o}}_i + \mathbf{J}_i^i \ddot{\mathbf{T}}_i \end{aligned} \quad (2.54)$$

where

$$\begin{aligned} \mathbf{e}_i &= \sum_{\lambda=i}^n (m_{\lambda} \ddot{\mathbf{o}}_{\lambda} + \mathbf{n}_{\lambda}^{\lambda} \mathbf{r} \ddot{\mathbf{T}}_{\lambda}) \\ \mathbf{e}_i &= \mathbf{e}_{i+1} + m_i \ddot{\mathbf{o}}_i + \mathbf{n}_i^i \mathbf{r} \ddot{\mathbf{T}}_i \end{aligned} \quad (2.55)$$

The $\ddot{\mathbf{T}}_{\lambda}$ term also has the same recursive expression as defined earlier in equation (2.32), though presently referring to a 3x3 rotation matrix. The $\ddot{\mathbf{o}}_{\lambda}$ term is given by:

$$\ddot{\mathbf{o}}_{\lambda} = \ddot{\mathbf{o}}_{\lambda-1} - \ddot{\mathbf{T}}_{\lambda} \ddot{\mathbf{o}}_{\lambda}^{\lambda-1} \quad (2.56)$$

This formulation decreases the complexity polynomial of the recursive Lagrangian formulation with 4x4 matrices by more than 50%. It requires:

1. $412n - 277$ multiplications, and
2. $320n - 201$ additions.

This translates into 2,195 multiplications and 1,719 additions, for a six degrees of freedom manipulator, which is well within the capacity of today's microprocessors.

D. RECURSIVE NEWTON-EULER'S MANIPULATOR DYNAMICS

Another possible formulation of the equations of motion is based on the Newton-Euler approach. While the Lagrangian dynamics were reworked with some effort into an efficient recursive form, this method, naturally, yields a set of recursive equations which can be applied to the links sequentially reducing the computational burden to its minimum possible.

In this derivation, each link is considered as a free body accelerating in space and obeying Newton's second law of dynamics for linear movement and Euler's equation for angular rotation.

Using the same notation as previously defined, the vectors \mathbf{o}_i (joint's i coordinate origin expressed in the base coordinate frame), \mathbf{o}_i^{i-1} (joint's i coordinate origin expressed in the $(i-1)^{th}$ coordinate frame), and \mathbf{o}_{i-1} (joint's $(i-1)$ coordinate origin expressed in the base coordinate frame) are related by:

$$\mathbf{o}_i = \mathbf{o}_{i-1} + \mathbf{o}_i^{i-1} \quad (2.57)$$

If \mathbf{v}_{i-1} and \mathbf{w}_{i-1} are, respectively, the linear and angular velocity of the coordinate system $(x_{i-1}, y_{i-1}, z_{i-1})$ with reference to the base coordinates, then the velocity of coordinate system (x_i, y_i, z_i) with reference to the base coordinates is:

$$\mathbf{v}_i = \frac{d\mathbf{o}_i^{i-1}}{dt} + \mathbf{w}_{i-1} \times \mathbf{o}_i^{i-1} + \mathbf{v}_{i-1} \quad (2.58)$$

Thus the acceleration is given by:

$$\dot{\mathbf{v}}_i = \frac{d^2\mathbf{o}_i^{i-1}}{dt^2} + \dot{\mathbf{w}}_{i-1} \times \mathbf{o}_i^{i-1} + 2\mathbf{w}_{i-1} \times \frac{d\mathbf{o}_i^{i-1}}{dt} + \mathbf{w}_{i-1} (\mathbf{w}_{i-1} \times \mathbf{o}_i^{i-1}) + \dot{\mathbf{v}}_{i-1} \quad (2.59)$$

in which $2\mathbf{w}_{i-1} \times \frac{d\mathbf{o}_i^{i-1}}{dt}$ is the Coriolis acceleration, $\mathbf{w}_{i-1} (\mathbf{w}_{i-1} \times \mathbf{o}_i^{i-1})$ is the centrifugal acceleration and \times denotes external product of vectors.

The angular velocity of the system (x_i, y_i, z_i) with reference to the base \mathbf{w}_i and its angular velocity with reference to the $(i-1)^{th}$ coordinate frame \mathbf{w}_i^{i-1} are related by:

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mathbf{w}_i^{i-1} \quad (2.60)$$

and

$$\dot{\mathbf{w}}_i = \dot{\mathbf{w}}_{i-1} + \dot{\mathbf{w}}_i^{i-1} \quad (2.61)$$

But,

$$\dot{\mathbf{w}}_i^{i-1} = \frac{d\mathbf{w}_i^{i-1}}{dt} + \mathbf{w}_{i-1} \times \mathbf{w}_i^{i-1} \quad (2.62)$$

Hence (2.61) becomes:

$$\dot{\mathbf{w}}_i = \dot{\mathbf{w}}_{i-1} + \frac{d\mathbf{w}_i^{i-1}}{dt} + \mathbf{w}_{i-1} \times \mathbf{w}_i^{i-1} \quad (2.63)$$

Since an angular motion of link i is about the z_{i-1} axis, then,

$$\mathbf{w}_i^{i-1} = \begin{cases} z_{i-1} \dot{q}_i & \text{if link } i \text{ is rotational} \\ 0 & \text{if link } i \text{ is translational} \end{cases} \quad (2.64)$$

Thus,

$$\frac{d\mathbf{w}_i^{i-1}}{dt} = \begin{cases} z_{i-1} \ddot{q}_i & \text{if link } i \text{ is rotational} \\ 0 & \text{if link } i \text{ is translational} \end{cases} \quad (2.65)$$

Combine (2.60), (2.63), (2.64), and (2.65) to yield:

$$\mathbf{w}_i = \begin{cases} \mathbf{w}_{i-1} + z_{i-1} \dot{q}_i & \text{if link } i \text{ is rotational} \\ \mathbf{w}_{i-1} & \text{if link } i \text{ is translational} \end{cases} \quad (2.66)$$

and

$$\dot{\mathbf{w}}_i = \begin{cases} \dot{\mathbf{w}}_{i-1} + z_{i-1} \ddot{q}_i + \mathbf{w}_{i-1} \times (z_{i-1} \dot{q}_i) & \text{if link } i \text{ is rotational} \\ \dot{\mathbf{w}}_{i-1} & \text{if link } i \text{ is translational} \end{cases} \quad (2.67)$$

Returning to equations (2.58) and (2.59), we note that if link i is translational in coordinates $(x_{i-1}, y_{i-1}, z_{i-1})$, it travels in the direction z_{i-1} , with joint velocity \dot{q}_i , relative to link $i-1$. If it is rotational in coordinates $(x_{i-1}, y_{i-1}, z_{i-1})$, it has an angular velocity \mathbf{w}_i^{i-1} . Thus,

$$\frac{d\mathbf{o}_i^{i-1}}{dt} = \begin{cases} \mathbf{w}_i^{i-1} \times \mathbf{o}_i^{i-1} & \text{if link } i \text{ is rotational} \\ z_{i-1} \dot{q}_i & \text{if link } i \text{ is translational} \end{cases} \quad (2.68)$$

$$\frac{d^2 \mathbf{o}_i^{i-1}}{dt^2} = \begin{cases} \frac{d\mathbf{w}_i^{i-1}}{dt} \times \mathbf{o}_i^{i-1} + \mathbf{w}_i^{i-1} \times (\mathbf{w}_i^{i-1} \times \mathbf{o}_i^{i-1}) & \text{if link } i \text{ is rotational} \\ \mathbf{z}_{i-1} \ddot{q}_i & \text{if link } i \text{ is translational} \end{cases} \quad (2.69)$$

Combine equations (2.60), (2.63), and (2.68) to yield:

$$\mathbf{v}_i = \begin{cases} \mathbf{w}_i \times \mathbf{o}_i^{i-1} + \mathbf{v}_{i-1} & \text{if link } i \text{ is rotational} \\ \mathbf{z}_{i-1} \dot{q}_i + \mathbf{w}_i \times \mathbf{o}_i^{i-1} + \mathbf{v}_{i-1} & \text{if link } i \text{ is translational} \end{cases} \quad (2.70)$$

$$\dot{\mathbf{v}}_i = \begin{cases} \dot{\mathbf{w}}_i \times \mathbf{o}_i^{i-1} + \mathbf{w}_i \times (\mathbf{w}_i \times \mathbf{o}_i^{i-1}) + \dot{\mathbf{v}}_{i-1} & \text{if link } i \text{ is rotational} \\ \mathbf{z}_{i-1} \ddot{q}_i + \dot{\mathbf{w}}_i \times \mathbf{o}_i^{i-1} + 2\mathbf{w}_i \times (\mathbf{z}_{i-1} \dot{q}_i) + \mathbf{w}_i \times (\mathbf{w}_i \times \mathbf{o}_i^{i-1}) + \dot{\mathbf{v}}_{i-1} & \text{if link } i \text{ is translational} \end{cases} \quad (2.71)$$

Equations (2.66), (2.67), (2.70), and (2.71) describe the recursive relations of velocities and accelerations between link $i-1$ and i . To derive the dynamic motion of the mechanical manipulator from the above kinematics information, each link is considered as a free body accelerating in space and obeying Newton's second law of dynamics:

$$\mathbf{F}_i = \frac{d(m_i \mathbf{v}_{c_i})}{dt} = m_i \mathbf{a}_{c_i}, \text{ for linear movement,} \quad (2.72)$$

and Euler's equation:

$$\mathbf{N}_i = \frac{d(\mathbf{I}_i \mathbf{w}_i)}{dt} = \mathbf{I}_i \dot{\mathbf{w}}_i + \mathbf{w}_i \times (\mathbf{I}_i \mathbf{w}_i), \text{ for angular rotation} \quad (2.73)$$

where,

\mathbf{F}_i is the total external vector force exerted on link i at the center of mass c_i ,

$\mathbf{v}_{c_i} = \frac{d\mathbf{c}_i}{dt}$ is the linear velocity of the center of mass c_i of link i ,

$\mathbf{a}_{c_i} = \frac{d\mathbf{v}_{c_i}}{dt}$ is the linear acceleration of the center of mass c_i of link i ,

\mathbf{N}_i is the total external vector moment exerted on link i at the center of mass c_i , and

\mathbf{I}_i is the inertia matrix of link i about its center of mass c_i with reference to the base coordinates system.

The other variables are as defined earlier.

The quantities w_i and \dot{w}_i can be computed from (2.66) and (2.67), while v_{c_i} and a_{c_i} can be derived from (2.58) and (2.59) as follows:

Replace o_i^{i-1} by c_i^i , o_i by c_i , and note that since both (x_i, y_i, z_i) and c_i^i are fixed on link i ,

$$\frac{do_i^{i-1}}{dt} = \frac{d^2 o_i^{i-1}}{dt^2} = 0.$$

Consequently, the equations describing v_{c_i} and a_{c_i} are:

$$v_{c_i} = w_{i-1} \times o_i^{i-1} + v_{i-1} \quad (2.74)$$

$$a_{c_i} = \dot{w}_{i-1} \times o_i^{i-1} + w_{i-1} \times (w_{i-1} \times o_i^{i-1}) + \dot{v}_{i-1} \quad (2.75)$$

The total external force F_i and moment N_i are those exerted on link i by gravity and neighboring links, that is,

$$F_i = f_i - f_{i+1} \quad (2.76)$$

$$N_i = n_i - n_{i+1} + (o_i - c_i) \times f_i - (o_{i+1} - c_i) \times f_{i+1} \\ N_i = n_i - n_{i+1} + (o_i - c_i) \times F_i - o_i^{i-1} \times f_{i+1} \quad (2.77)$$

where

f_i is the force exerted on link i by link $i-1$, and

n_i is the moment exerted on link i by link $i-1$.

Since $c_i - o_{i-1} = o_i^{i-1} + c_i^i$, equations (2.76) and (2.77) can be expressed in recursive relations as

$$f_i = f_{i+1} + F_i \quad (2.78)$$

$$n_i = n_{i+1} + o_i^{i-1} \times f_{i+1} + (o_i^{i-1} + c_i^i) \times F_i + N_i \quad (2.79)$$

According to the convention for establishing link coordinate systems for a mechanical manipulator, the motion of link i may only be either a rotation in the coordinate system $(x_{i-1}, y_{i-1}, z_{i-1})$ about z_{i-1} axis, or a translation relative to the coordinate system $(x_{i-1}, y_{i-1}, z_{i-1})$ along z_{i-1} . Therefore, if the joint i is rotational, the input torque τ_i at that joint is the sum of the projection of n_i onto the z_{i-1} axis and the viscous damping

moment in that coordinate system. If, however, the joint i is translational, the input force τ_i at that joint is the sum of the projection of f_i onto the z_{i-1} axis and the viscous damping force in that coordinate system. That is,

$$\tau_i = \begin{cases} n_i^T z_{i-1} + b_i \dot{q}_i & \text{if link } i \text{ is rotational} \\ f_i^T z_{i-1} + b_i \dot{q}_i & \text{if link } i \text{ is translational} \end{cases} \quad (2.80)$$

where b_i is the viscous damping coefficient for joint i .

In summary, the complete set of equations of motion for the mechanical manipulator with n joints and $n+1$ links consists of equations (2.66), (2.67), (2.70) through (2.75) and (2.78) through (2.80) for $i=1,2,\dots,n$. Unfortunately, because these equations are referenced to the base coordinate systems, the inertia matrix I_i is dependent on the changing orientation of link i , which complicates the computation. A more efficient technique for computing the joint input forces and torques is to have each link's dynamics referenced to its own link coordinates [68]. This may easily be achieved using the rotation matrix R_i^{i-1} defined earlier and noting that since each coordinate system is orthogonal, then:

$$(R_i^{i-1})^{-1} = (R_i^{i-1})^T = R_{i-1}^i \quad (2.81)$$

Instead of computing w_i , \dot{w}_i , v_i , a_{c_i} , F_i , N_i , f_i , n_i , and τ_i , compute $R^i w_i$, $R^i \dot{w}_i$, $R^i v_i$, $R^i a_{c_i}$, $R^i F_i$, $R^i N_i$, $R^i f_i$, $R^i n_i$, and $R^i \tau_i$. Hence the complete set of equations of motion becomes:

$$R^i w_i = \begin{cases} R_{i-1}^i (R^{i-1} w_{i-1} + z_0 \dot{q}_i) & \text{if link } i \text{ is rotational} \\ R_{i-1}^i (R^{i-1} w_{i-1}) & \text{if link } i \text{ is translational} \end{cases} \quad (2.82)$$

$$R^i \dot{w}_i = \begin{cases} R_{i-1}^i [R^{i-1} \dot{w}_{i-1} + z_0 \ddot{q}_i + (R^{i-1} w_{i-1}) \times z_0 \dot{q}_i] & \text{if link } i \text{ is rotational} \\ R_{i-1}^i (R^{i-1} \dot{w}_i) & \text{if link } i \text{ is translational} \end{cases} \quad (2.83)$$

$$R^i \dot{v}_i = \begin{cases} (R^i \dot{w}_i) \times (R^i o_i^{i-1}) + R_{i-1}^i (R^{i-1} \dot{v}_{i-1}) \\ + (R^i w_i) \times [(R^i w_i) \times (R^i o_i^{i-1})] & \text{if link } i \text{ is rotational} \\ R_{i-1}^i (z_0 \ddot{q}_i + R^{i-1} \dot{v}_{i-1}) + (R^i \dot{w}_i) \times (R^i o_i^{i-1}) \\ + 2(R^i w_i) \times (R_{i-1}^i z_0 \dot{q}_i) + (R^i w_i) \\ \times [(R^i w_i) \times (R^i o_i^{i-1})] & \text{if link } i \text{ is translational} \end{cases} \quad (2.84)$$

$$R^i a_{c_i} = (R^i \dot{w}_i) \times (R^i c_i^i) + (R^i w_i) \times [(R^i w_i) \times (R^i c_i^i)] + R^i \dot{v}_i \quad (2.85)$$

$$R^i F_i = m_i R^i a_{c_i} \quad (2.86)$$

$$R^i N_i = (R^i I_i R_i)(R^i \dot{w}_i) + (R^i w_i) \times [(R^i I_i R_i)(R^i w_i)] \quad (2.87)$$

$$R^i f_i = R_{i+1}^i (R^{i+1} f_{i+1}) + R^i F_i \quad (2.88)$$

$$R^i n_i = R_{i+1}^i [R^{i+1} n_{i+1} + (R^{i+1} o_i^{i-1}) \times (R^{i+1} f_{i+1})] \\ + (R^i o_i^{i-1} + R^i c_i^i) \times (R^i F_i) + (R^i N_i) \quad (2.89)$$

$$\tau_i = \begin{cases} (R^i n_i)^T (R_{i-1}^i z_0) + b_i \dot{q}_i & \text{if link } i \text{ is rotational} \\ (R^i f_i)^T (R_{i-1}^i z_0) + b_i \dot{q}_i & \text{if link } i \text{ is translational} \end{cases} \quad (2.90)$$

This formulation gives a 60% reduction in computation over the recursive Lagrangian formulation. It requires:

1. $150n - 48$ multiplications, and
2. $131n - 48$ additions.

E. CONCLUSIONS

In this chapter alternative formulations for deriving the equations of motion of serial link manipulators have been described. The emphasis has been put on real time computational complexity in terms of required mathematical operations per trajectory set point. One should not, however, be misled by the fact that in the above development, the recursive Newton-Euler equations are almost three times more efficient than the recursive

Lagrangian equations. The discrepancy between the two formulations is due to the difference in the angular velocity vector representation used by each method. In the Newton–Euler formulation, the angular velocity is adequately represented with a 3x1 vector \mathbf{w}_i , whereas in the Lagrangian formulation, it is redundantly represented with a 3x3 matrix U_{ij} . A factor of three in the relative efficiency of the two formulations is therefore to be expected. The redundancy in the angular velocity representation is manifested in the rotational kinetic energy expression. For the 3x3 representation, the rotational kinetic energy is as derived in equation (2.15):

$$K_i = \frac{1}{2} \text{Tr} \left[\left(\sum_{\lambda=1}^i U_{i\lambda} \dot{q}_\lambda \right) J_i^i \left(\sum_{k=1}^i U_{ik}^T \dot{q}_k \right) \right]$$

whereas for the 3x1 representation, the rotational kinetic energy is:

$$K_i = \frac{1}{2} \mathbf{w}_i^T \mathbf{I}_i \mathbf{w}_i \quad (2.91)$$

Using this new representation of the rotational kinetic energy, the complete generalized force expression τ_i in the Lagrangian equation (2.22) changes to:

$$\tau_i = \sum_{\lambda=i}^n \left[m_\lambda (\mathbf{g} + \ddot{\mathbf{c}}_\lambda) \cdot \frac{\partial \dot{\mathbf{c}}_\lambda}{\partial \dot{\mathbf{q}}_i} + [\mathbf{I}_\lambda \dot{\mathbf{w}}_\lambda + \mathbf{w}_\lambda \times (\mathbf{I}_\lambda \mathbf{w}_\lambda)] \cdot \frac{\partial \mathbf{w}_\lambda}{\partial \dot{\mathbf{q}}_i} \right] \quad (2.92)$$

or

$$\tau_i = \sum_{\lambda=i}^n \left[\mathbf{f}_\lambda \cdot \frac{\partial \dot{\mathbf{c}}_\lambda}{\partial \dot{\mathbf{q}}_i} + \mathbf{n}_\lambda \cdot \frac{\partial \mathbf{w}_\lambda}{\partial \dot{\mathbf{q}}_i} \right] \quad (2.93)$$

where \mathbf{f}_λ represents the net force in Newton's equation, and \mathbf{n}_λ represents the net torque in Euler's equation.

Therefore, contrary to what might have appeared earlier, there is no difference in the computational complexity of dynamics formulations derived from the Newton–Euler equations or the Lagrange equations. The recursive Newton–Euler equations are no more efficient than the recursive Lagrangian equations as long as the same representation of angular velocity is used. Moreover, the Newton–Euler equations would become as

inefficient as the original Uicker/Kahn equations if they were expressed in closed form.

Consequently, the emphasis on computational complexity or on advanced control strategies synthesis should rest on the structure of the computation rather than on the derivation from Lagrange versus Newton—Euler equations.

In addition, the designer will probably need both structures of the dynamic equations and more than one method of obtaining these equations throughout the different phases of the design process. He will need:

1. A closed form expression of the manipulator dynamics in the early stages of the design process in order to be able to synthesize adequate control laws,
2. More than one method of deriving the system dynamics equations in the computer simulation phase in order to be able to compare the solution obtained by different methods and place greater confidence on the simulation program, and
3. A recursive form expression of the manipulator dynamics when implementing the chosen control law in real time.

III. ADAPTIVE CONTROL FOR MECHANICAL MANIPULATORS

A. INTRODUCTION

The problem of controlling articulated mechanical systems such as manipulators using conventional control methods is very difficult when high speed and high accuracy operations are desired. The difficulty arises from the fact that such linkages are characterized by highly nonlinear and coupled ordinary differential equations. Closed form analytical solutions to these differential equations are not available. Instead, they must be solved by numerical integration on a digital computer, which, on the other hand, imposes a serious limitation on the number of calculations that can be performed in real time.

The problem becomes even more difficult when the plant parameters are not precisely known and vary in time, as in most robotics applications. Furthermore, a joint angles to end point coordinates matrix transformation are usually required in such systems, which increases the burden on the computing machine.

To maintain good performance over a wide range of motions and payloads, researchers have turned to adaptive control methods for their ability to adjust to parameters uncertainties and load disturbances. Unfortunately, most of these methodologies are not computationally efficient. As we have indicated in Chapter 1, two approaches to adaptive control theory can be found in the literature.

In the Model Reference Adaptive Control scheme, the manipulator dynamic model is not directly used in the design so that the on line solution of differential equations is not required in the implementation. The manipulator is controlled by adjusting position and velocity feedback gains to follow a prescribed reference model.

In the Learning Model Adaptive Control method, a model of the plant is obtained by on line parameter estimation techniques. This estimated model is then used in the feedback

control. For reasons of computation efficiency, most of the techniques in this category are based on linearized models of the manipulator which constrain the range of validity and of acceptable performances. Our main focus in this chapter will be to develop an algorithm for the Model Reference Adaptive Control of mechanical manipulators.

In the next section, we will formulate the dynamic equations for mechanical manipulators in state space. This representation is more suitable for analyzing the performances of such methodologies. We will also give a detailed derivation of the state space equations of a two link arm model for illustration.

Section three will review some leading adaptive control methods. These techniques will be simulated on the two link model using IBM/DSL [69]. We will compare their performances and point out some of their advantages as well as some of their limitations. The aim is to show where the need for better adaptive control strategies is felt.

In section four, a novel adaptive control law which guarantees global stability and yields better performances is synthesized. A rapprochement between this method and the Model Reference Adaptive Control methodologies is also established.

Section five summarizes the main results obtained in this study and outlines some areas for future research.

B. MECHANICAL MANIPULATOR DYNAMICS IN TERMS OF STATE VARIABLES

The standard inverse dynamics equations describing a mechanical manipulator are given in equation (2.24) of Chapter 2 and are reproduced here for convenience.

$$\tau(t) = \mathbf{A}[\mathbf{q}(t)]\ddot{\mathbf{q}}(t) + \mathbf{V}[\mathbf{q}(t),\dot{\mathbf{q}}(t)] + \mathbf{G}[\mathbf{q}(t)] \quad (3.1)$$

These equations can be rewritten as:

$$\ddot{\mathbf{q}}(t) = \mathbf{A}^{-1}[\mathbf{q}(t)] \left[\tau(t) - \mathbf{V}[\mathbf{q}(t),\dot{\mathbf{q}}(t)] - \mathbf{G}[\mathbf{q}(t)] \right] \quad (3.2)$$

In order to be able to gain better analytical insight to the control system design, it is convenient to rewrite equation (3.2) in terms of state variables.

Naturally, if one has no particular reason for other choices, the state variables and the input control vectors are, respectively, defined as:

$$\mathbf{X}_p(t) = \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix} \quad (3.3)$$

$$\mathbf{U}_p(t) = \tau(t) \quad (3.4)$$

where, $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ and $\tau(t)$ are as defined earlier in Chapter 2.

In addition, the vectors $\mathbf{V}[\mathbf{q}(t), \dot{\mathbf{q}}(t)]$ and $\mathbf{G}[\mathbf{q}(t)]$ can be expressed as:

$$\mathbf{V}[\mathbf{q}(t), \dot{\mathbf{q}}(t)] = \mathbf{V}_1[\mathbf{q}(t), \dot{\mathbf{q}}(t)] \dot{\mathbf{q}}(t) \quad (3.5)$$

$$\mathbf{G}[\mathbf{q}(t)] = \mathbf{G}_1[\mathbf{q}(t)] \mathbf{q}(t) \quad (3.6)$$

Therefore, equation (3.2) can be rewritten as:

$$\dot{\mathbf{X}}_p(t) = \mathbf{A}_p(\mathbf{X}_p(t), t) \mathbf{X}_p(t) + \mathbf{B}_p(\mathbf{X}_p(t), t) \mathbf{U}_p(t) \quad (3.7)$$

where

$$\dot{\mathbf{X}}_p(t) = \frac{d}{dt}(\mathbf{X}_p(t)) = \begin{bmatrix} \dot{\mathbf{q}}(t) \\ \ddot{\mathbf{q}}(t) \end{bmatrix} \quad (3.8)$$

$$\mathbf{A}_p(\mathbf{X}_p(t), t) = \begin{bmatrix} \mathbf{0} & \vdots & \mathbf{I}_n \\ \dots\dots\dots & \vdots & \dots\dots\dots \\ \mathbf{A}^{-1}(\mathbf{q}(t)) \mathbf{G}_1(\mathbf{q}(t)) & \vdots & -\mathbf{A}^{-1}(\mathbf{q}(t)) \mathbf{V}_1(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \end{bmatrix} \quad (3.9)$$

$$\mathbf{B}_p(\mathbf{X}_p(t), t) = \begin{bmatrix} \mathbf{0} \\ \dots\dots\dots \\ \mathbf{A}^{-1}(\mathbf{q}(t)) \end{bmatrix} \quad (3.10)$$

Here \mathbf{I}_n is the identity matrix of order n .

The matrices $\mathbf{V}_1[\mathbf{q}(t), \dot{\mathbf{q}}(t)]$ and $\mathbf{G}_1[\mathbf{q}(t)]$ are found as follows:

Notice that the expression of $\mathbf{V}[\mathbf{q}(t), \dot{\mathbf{q}}(t)]$ given in equation (2.22) of Chapter 2 can be rewritten as:

$$\mathbf{V}[\mathbf{q}(t), \dot{\mathbf{q}}(t)] = \mathbf{V}_2[\mathbf{q}(t)] \mathbf{V}_3[\dot{\mathbf{q}}(t)] \quad (3.11)$$

where $\mathbf{V}_3[\dot{\mathbf{q}}(t)]$ is given by:

$$V_3[\dot{q}(t)] = \begin{bmatrix} \dot{q}_1(t)\dot{q}_1(t) \\ \dot{q}_1(t)\dot{q}_2(t) \\ \vdots \\ \dot{q}_1(t)\dot{q}_n(t) \\ \vdots \\ \dot{q}_n(t)\dot{q}_1(t) \\ \vdots \\ \dot{q}_n(t)\dot{q}_n(t) \end{bmatrix}$$

and $V_2[q(t)]$ is a matrix containing all the remaining terms of $V[q(t), \dot{q}(t)]$.

$V_3[\dot{q}(t)]$ can be expressed as:

$$V_3[\dot{q}(t)] = \begin{bmatrix} \dot{q}_1(t)\dot{q}_1(t) \\ \dot{q}_1(t)\dot{q}_2(t) \\ \vdots \\ \dot{q}_1(t)\dot{q}_n(t) \\ \vdots \\ \dot{q}_n(t)\dot{q}_1(t) \\ \vdots \\ \dot{q}_n(t)\dot{q}_n(t) \end{bmatrix} = V_4[\dot{q}(t)]\dot{q}(t) \quad (3.12)$$

with

$$V_4[\dot{q}(t)] = \begin{bmatrix} \dot{q}_1(t) & 0 & \dots & 0 \\ 0 & \dot{q}_1(t) & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \dot{q}_1(t) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \dot{q}_n(t) & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \dot{q}_n(t) \end{bmatrix} \quad (3.12)$$

Therefore,

$$V_1[q(t), \dot{q}(t)] = V_2[q(t)]V_4[\dot{q}(t)] \quad (3.13)$$

To obtain $G_1[q(t)]$, let

$$G_1^I[q(t)] = \frac{\|G[q(t)]\|}{\|q(t)\|} q(t) \quad (3.14)$$

$$G_1^2[q(t)] = \frac{G[q(t)] - G_1^1[q(t)]}{\|G[q(t)] - G_1^1[q(t)]\|} \quad (3.15)$$

Hence,

$$G_1[q(t)] = H[G_1^2[q(t)]] \frac{\|G[q(t)]\|}{\|q(t)\|} \quad (3.16)$$

where $\|\cdot\|$ is the Euclidean norm of a vector and $H[\cdot]$ is the Householder transformation [70] defined as:

$$H[u] = I - 2.u.u^T \quad (3.17)$$

C. SIMULATION STUDY OF SOME ADAPTIVE CONTROL METHODS

In the remainder of this chapter, a two revolute joints arm model is considered. This model is shown in Figure 3.1. The equations of motions of this mechanical system are derived in detailed in the Appendix. Based on this model, a new adaptive algorithm, as well as some of the strategies presented in Chapter 1, are studied here in more detail. Considered are the inverse dynamics control technique, the variable structure control approach, the model following strategy and the perturbation control theory. The basic idea behind all these methodologies is to synthesize a control input τ which will force the robot to follow the output of a reference model. The behavior we are concerned with here is the tracking in real time of desired trajectories. The reference model can be either a stable linear time invariant decoupled system as in the Adaptive Linear Model Following Control, or a combination of models such as in the Variable Structure Control, or a nonlinear nominal model of the plant as in the inverse dynamics control and the perturbation theory.

1. Inverse Dynamics Technique

In this technique, the control input $U_p(t)$ is chosen as:

$$U_p(t) = \hat{A}[q(t)] \left\{ \ddot{q}^d(t) + K_v \dot{e}(t) + K_p e(t) \right\} + \hat{V}[q(t), \dot{q}(t)] + \hat{G}[q(t)] \quad (3.18)$$

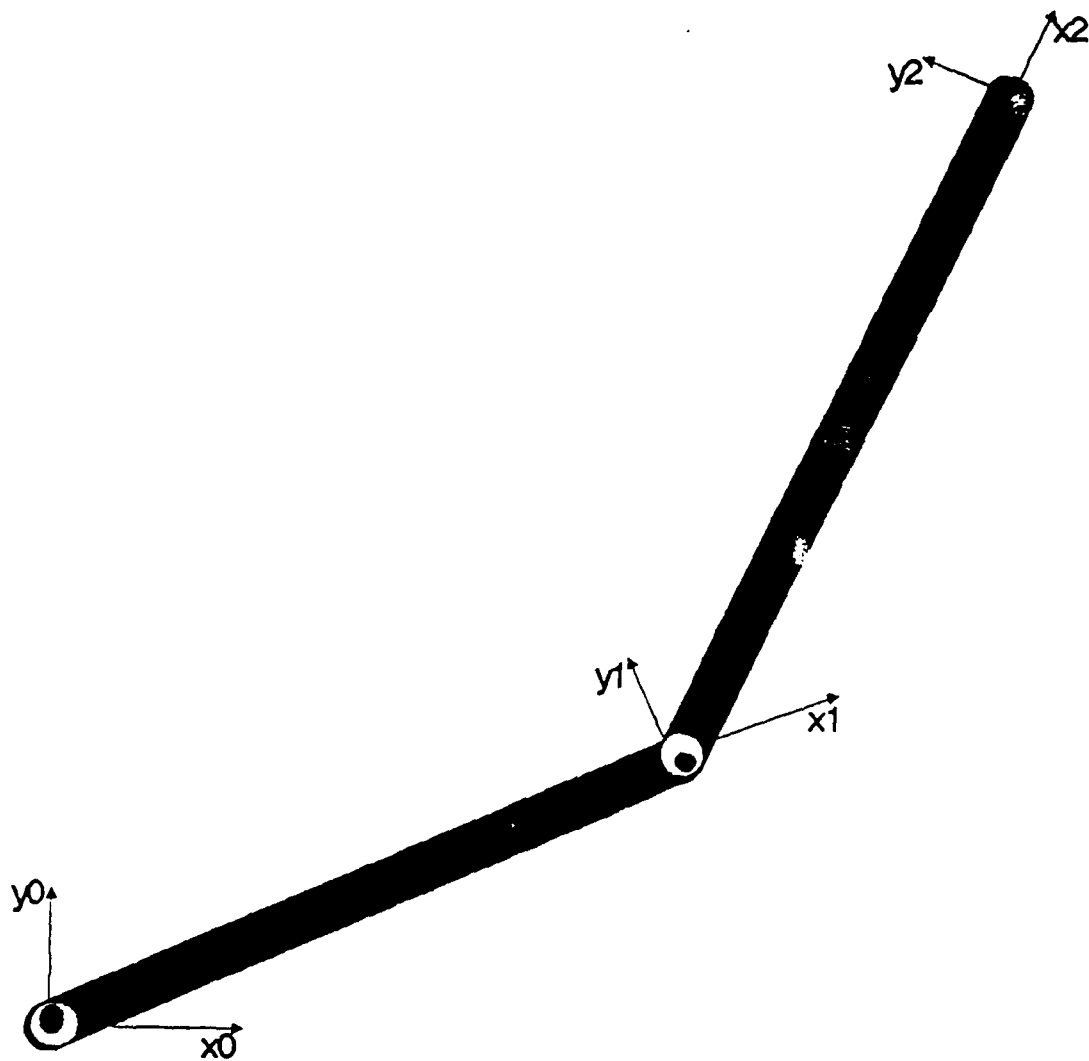


Figure 3.1: A two link Mechanical Manipulator

where $\hat{A}[q(t)]$, $\hat{V}[q(t), \dot{q}(t)]$ and $\hat{G}[q(t)]$ are some estimates of $A[q(t)]$, $V[q(t), \dot{q}(t)]$ and $G[q(t)]$, respectively.

This control law consist of two basic loops:

1. A feedforward component:

$$U_{pf}(t) = \hat{A}[q(t)] \ddot{q}^d(t) + \hat{V}[q(t), \dot{q}(t)] + \hat{G}[q(t)].$$

This component is based on a dynamic model of the manipulator. It compensates for the interaction forces among various joints.

2. A feedback component:

$$U_{pb}(t) = \hat{A}[q(t)] \{ K_v \dot{e}(t) + K_p e(t) \}.$$

This component is based on position and derivative feedback. It computes the necessary correction torques to compensate for any deviations from the desired trajectory.

Among the attractive features of this method is the fact that, in principle, it turns a nonlinear, coupled mechanical system into a linear, decoupled, and stable system. This can be seen by substituting the above torques expression of equation (3.18) into equation (3.1) to obtain:

$$A \ddot{q} + V + G = \hat{A} \{ \ddot{q}^d + K_v \dot{e} + K_p e \} + \hat{V} + \hat{G} \quad (3.19)$$

where the arguments have been omitted for convenience.

If $\hat{A} = A$, $\hat{V} = V$ and $\hat{G} = G$, equation (3.19) reduces to:

$$A[q(t)] \{ \ddot{e}(t) + K_v \dot{e}(t) + K_p e(t) \} = 0 \quad (3.20)$$

Since $A[q(t)]$ is always nonsingular, K_p and K_v can be appropriately chosen so that the position error vector $e(t)$ approaches zero asymptotically. This control strategy is simulated on the two arm model presented earlier. The desired trajectories are chosen as:

$$q_1^d(t) = 270t^2 - 180t^3 \quad (3.21)$$

$$q_2^d(t) = 45 + 270t^2 - 180t^3 \quad (3.22)$$

These trajectories are shown in Figures 3.2 and 3.3. The gain matrices K_p and K_v are chosen as:

$$\begin{aligned} K_p &= \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix} \\ K_v &= \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix} \end{aligned} \quad (3.23)$$

In this case, where we assume complete and exact knowledge of the manipulator dynamics, we obtain perfect tracking of the desired trajectories. This can be seen in Figures 3.4 and 3.5, where the actual trajectories (q_1 and q_2) and the desired trajectories (q_1^d and q_2^d) are virtually indistinguishable with the tracking errors (e_1 and e_2) shown in Figure 3.6. Also the magnitudes of the torques applied at the joint actuators are bounded with reasonable values as shown in Figure 3.7.

The main drawback of this method is inherent in its assumption that one can accurately compute the counterparts of $A[q(t)]$, $V[q(t), \dot{q}(t)]$ and $G[q(t)]$. Unfortunately, this is not always the case in robotics applications. When $\hat{A}[q(t)]$, $\hat{V}[q(t), \dot{q}(t)]$ and $\hat{G}[q(t)]$ are not equal to $A[q(t)]$, $V[q(t), \dot{q}(t)]$ and $G[q(t)]$, the quality of the tracking degrades and the system may even become unstable. Simulation results with 10% error in the load are reported in Figures 3.8 through 3.11. In Figures 3.8 and 3.9, we can see that the actual trajectories (q_1 and q_2) diverge from the desired trajectories (q_1^d and q_2^d). The position error is in the order of 6° in the first link and of 13° in the second link (Figure 3.10). The applied torques (Figure 3.11) are of reasonable values.

One may try to overcome this limitation by combining the above scheme with an on line identification algorithm to compute $\hat{A}[q(t)]$, $\hat{V}[q(t), \dot{q}(t)]$, and $\hat{G}[q(t)]$ [71]. This, however, is computationally demanding since the computed torque, in itself, requires a number $O(n^4)$ of calculations, n being the number of links in the manipulator.

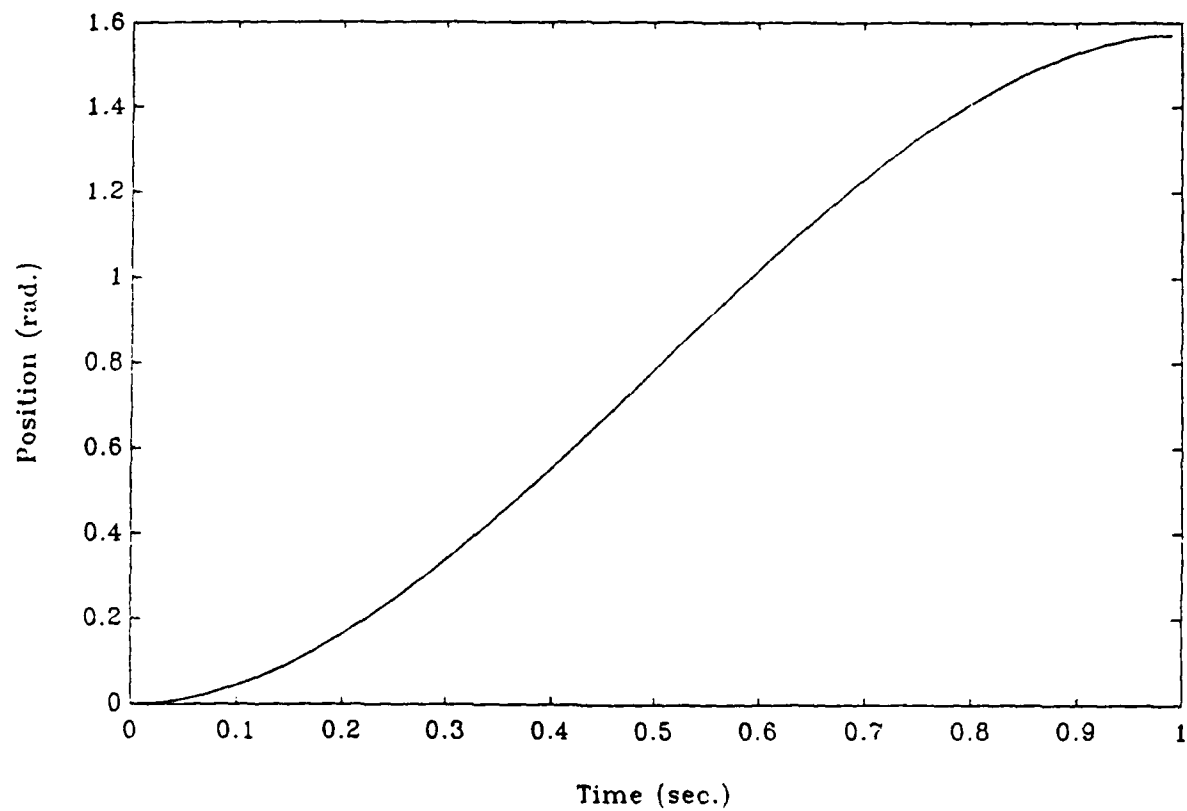


Figure 3.2: The first link test trajectory

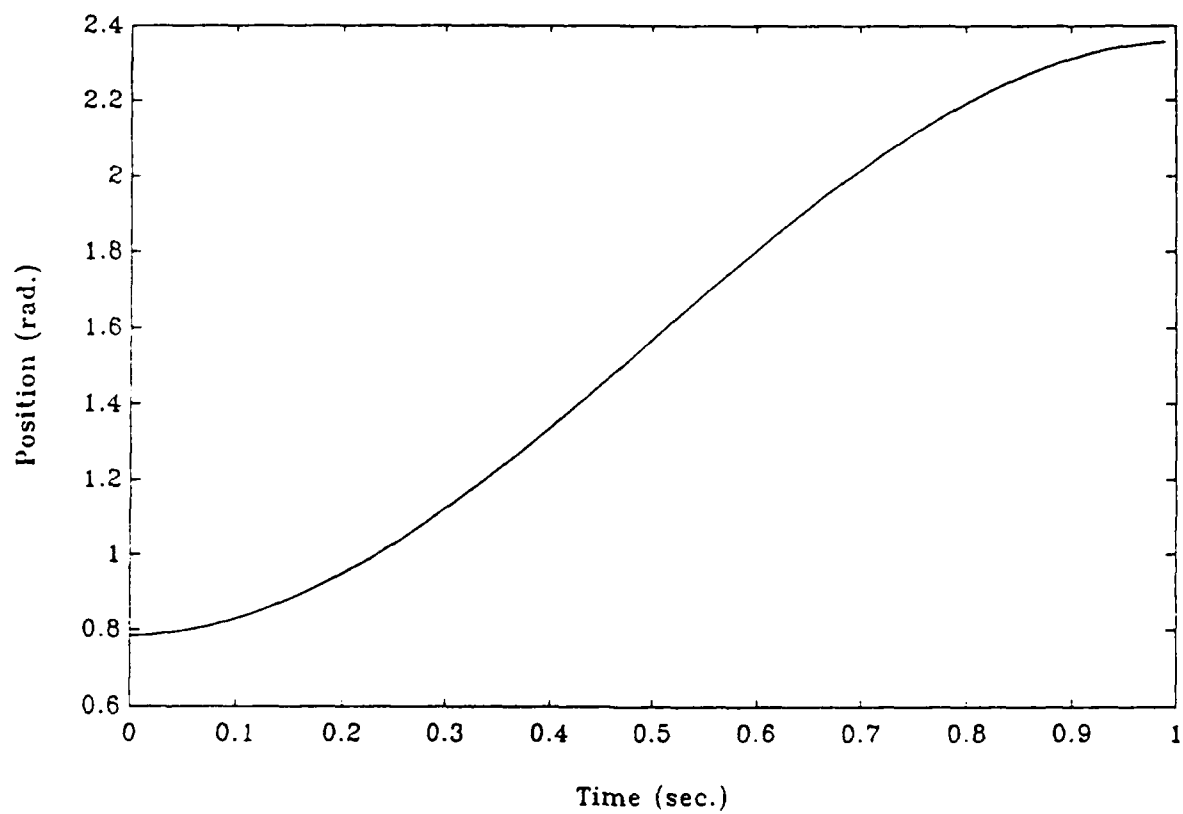


Figure 3.3: The second link test trajectory

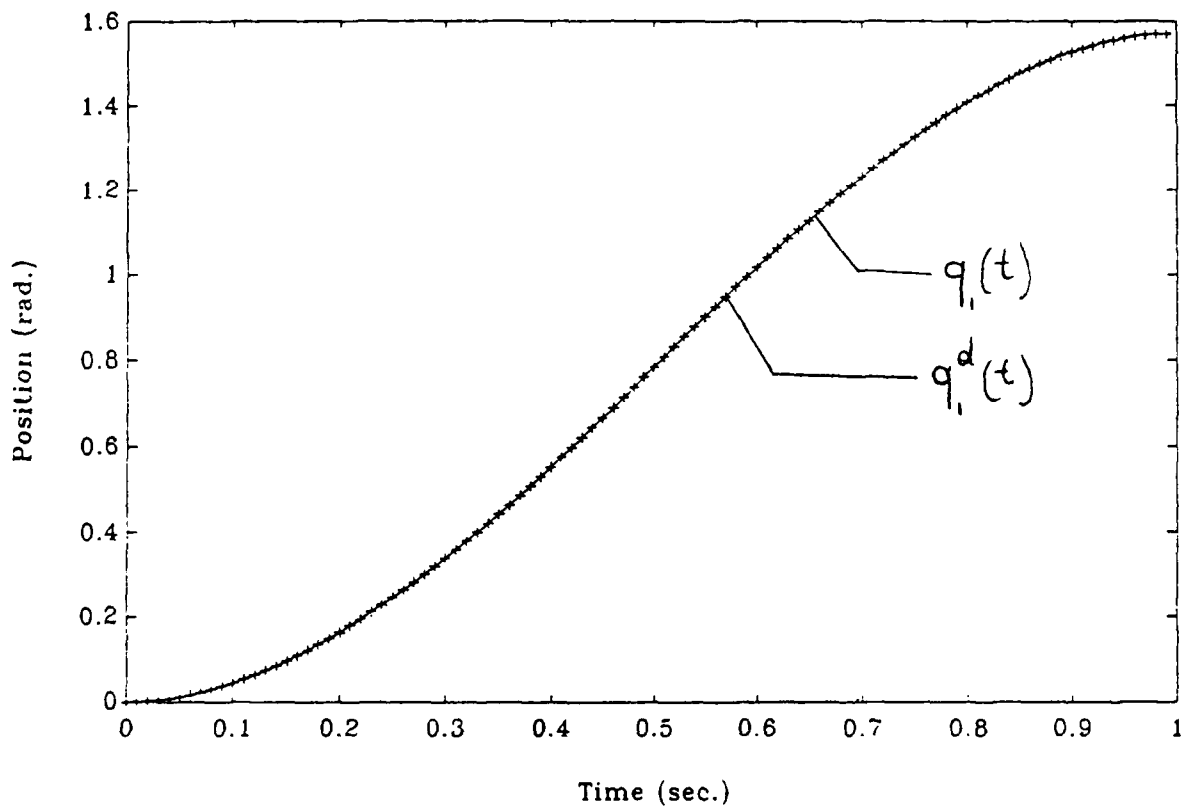


Figure 3.4: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Inverse Dynamics Controller (Perfect Modeling)

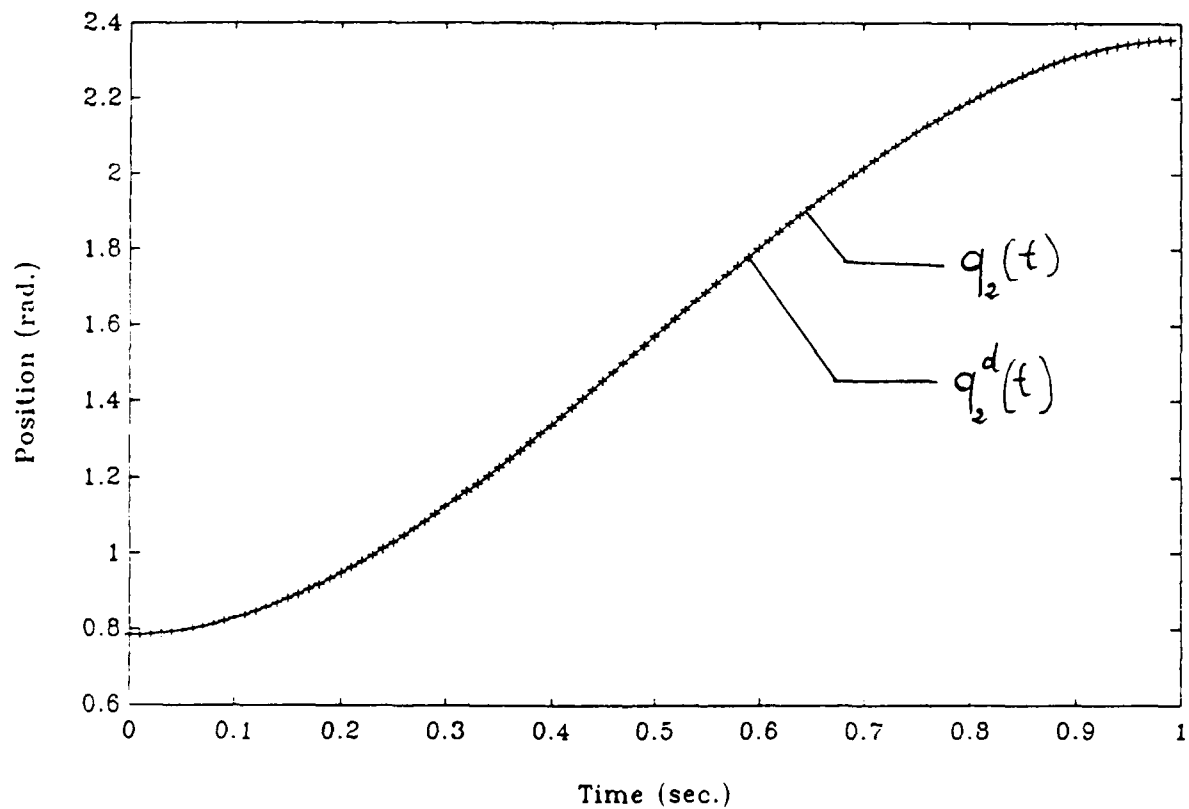


Figure 3.5: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Inverse Dynamics Controller (Perfect Modeling)

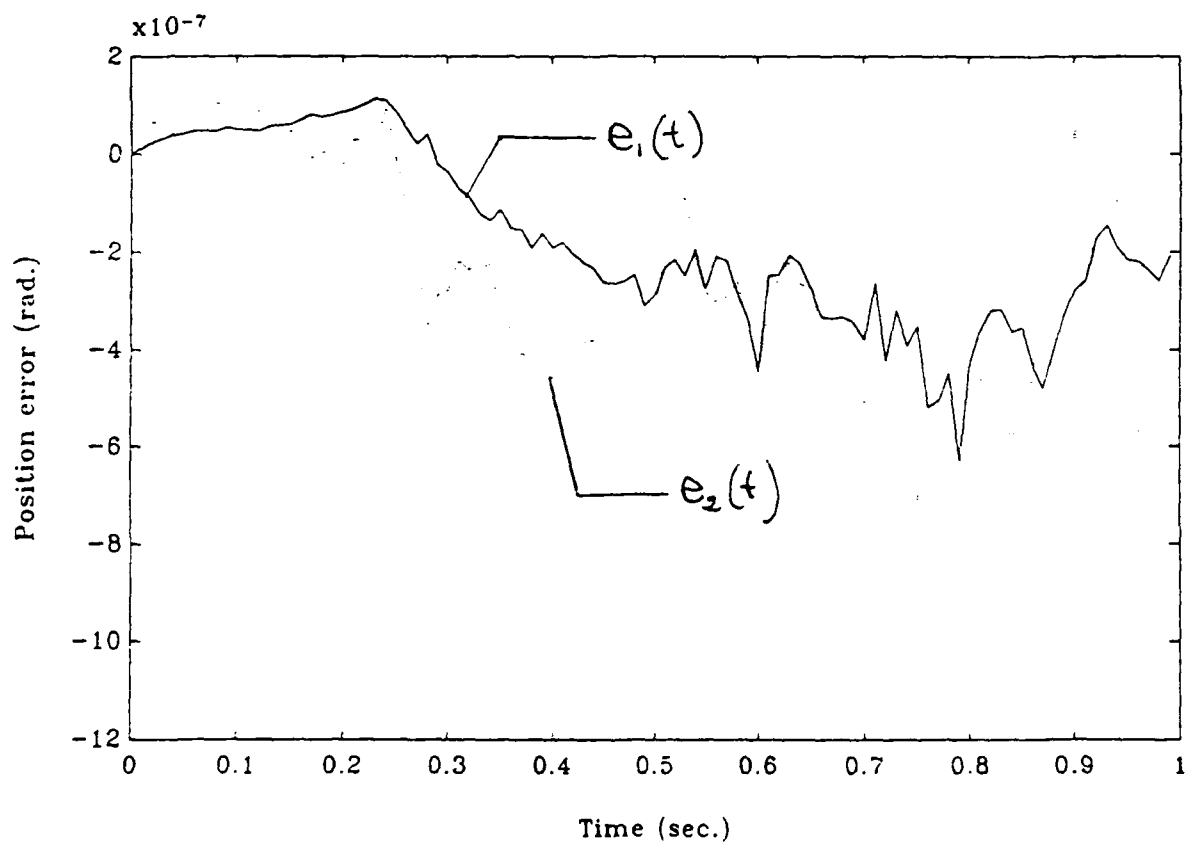


Figure 3.6: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Inverse Dynamics Controller (Perfect Modeling)

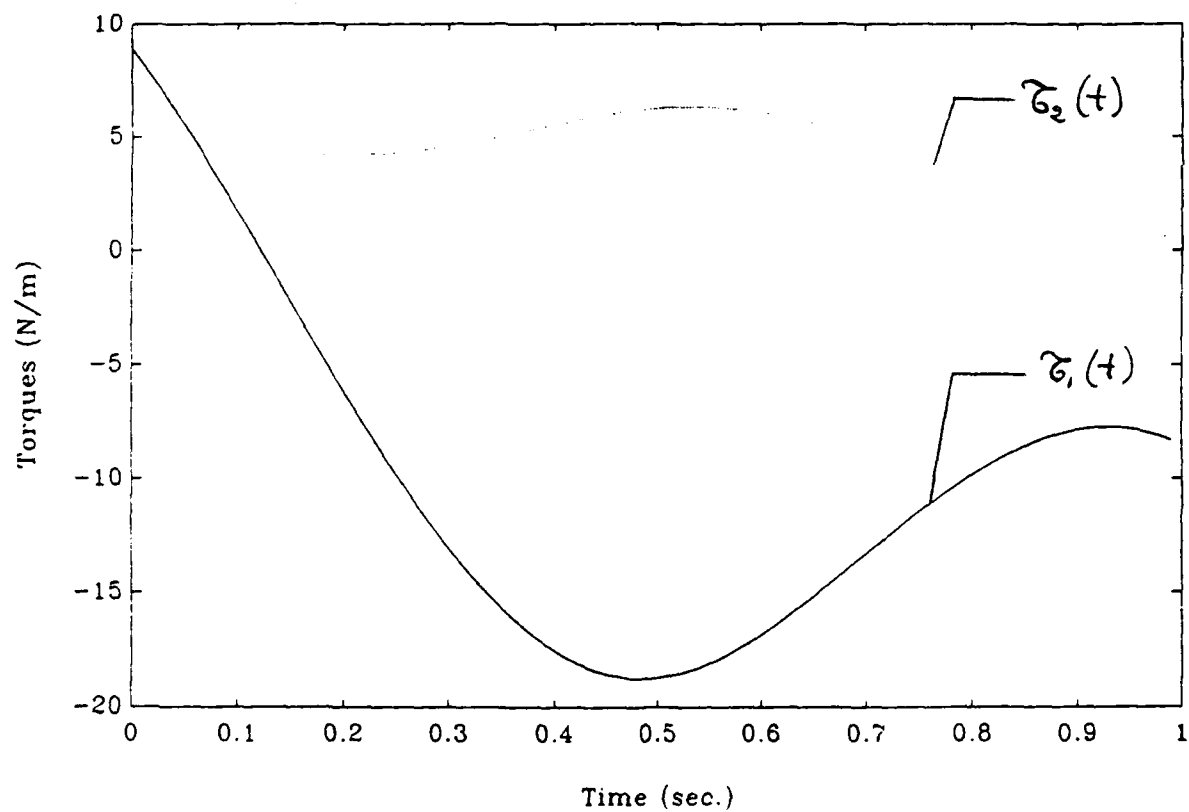


Figure 3.7: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Inverse Dynamics Controller (Perfect Modeling)

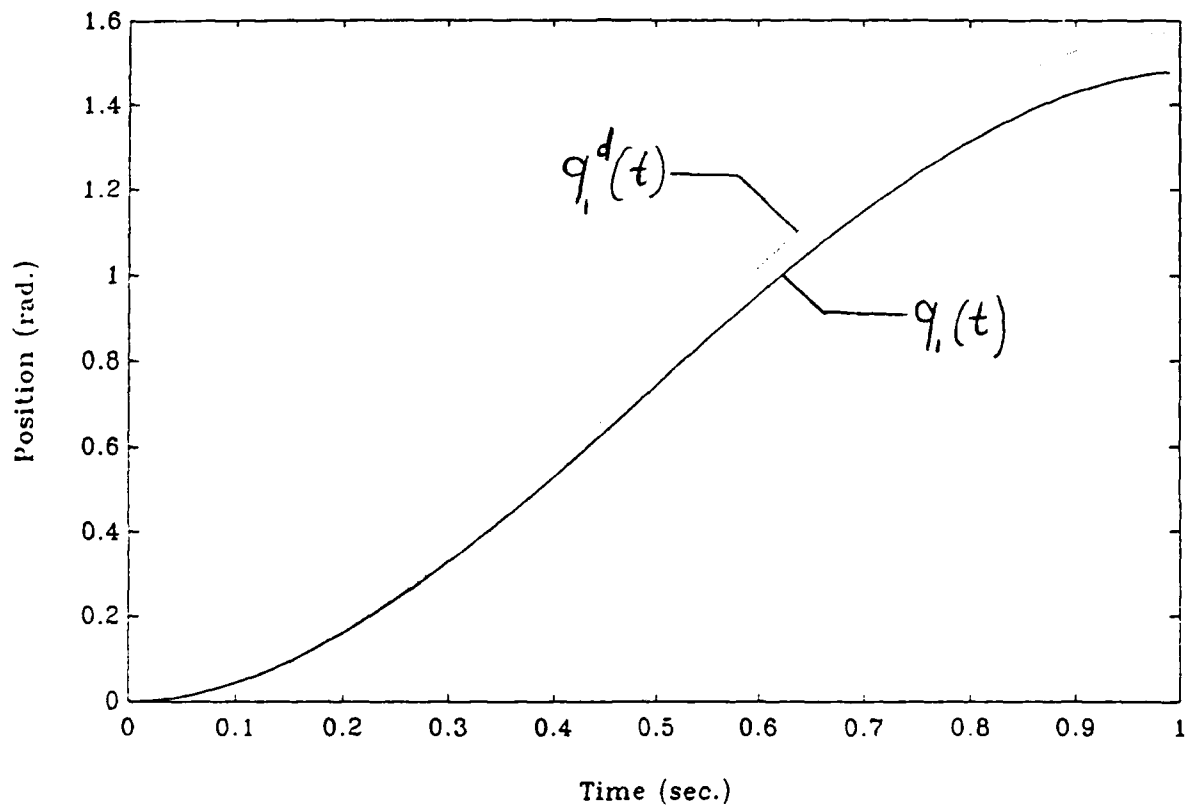


Figure 3.8: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Inverse Dynamics Controller (Approximate Modeling)

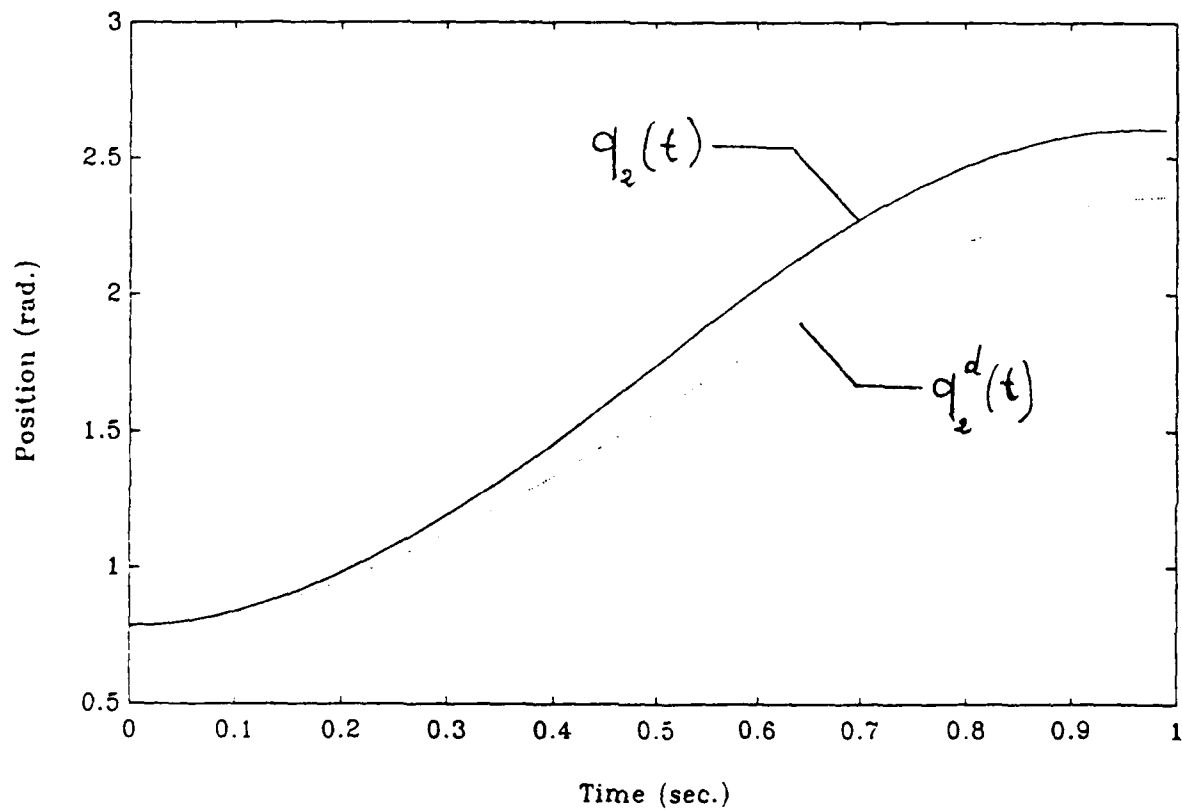


Figure 3.9: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Inverse Dynamics Controller (Approximate Modeling)

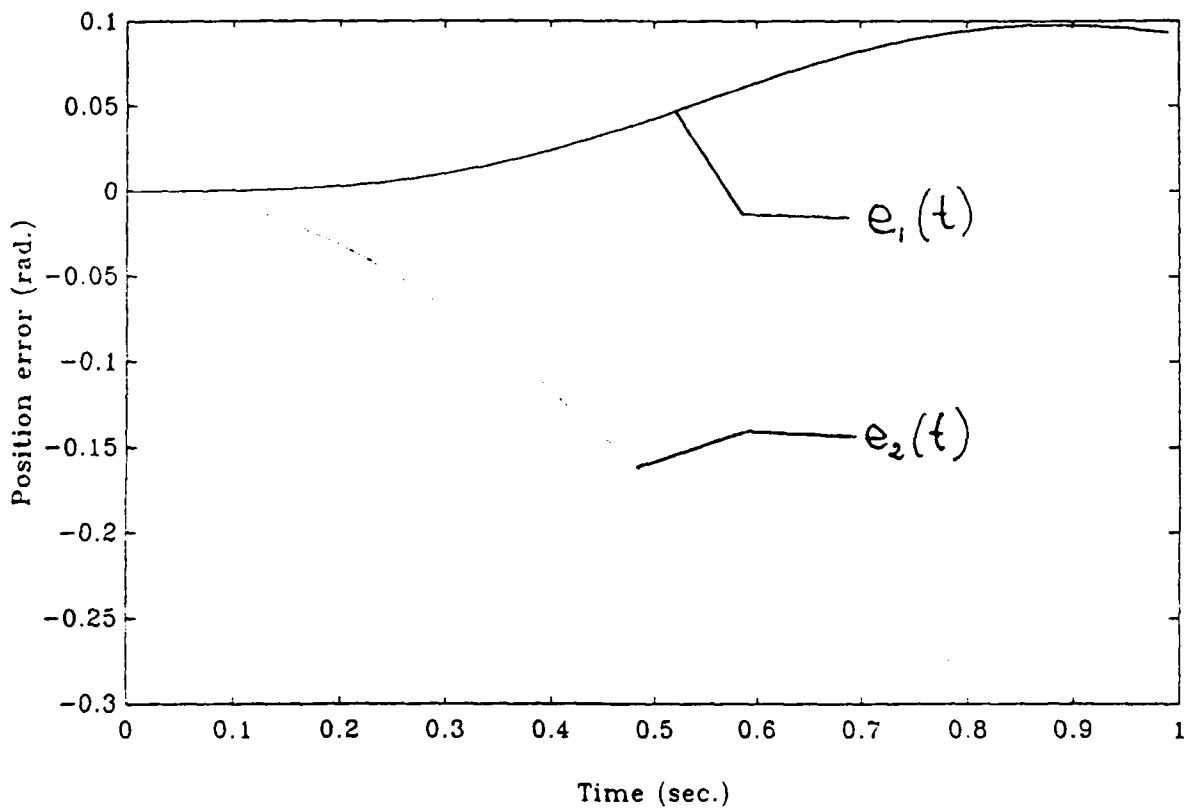


Figure 3.10: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Inverse Dynamics Controller (Approximate Modeling)

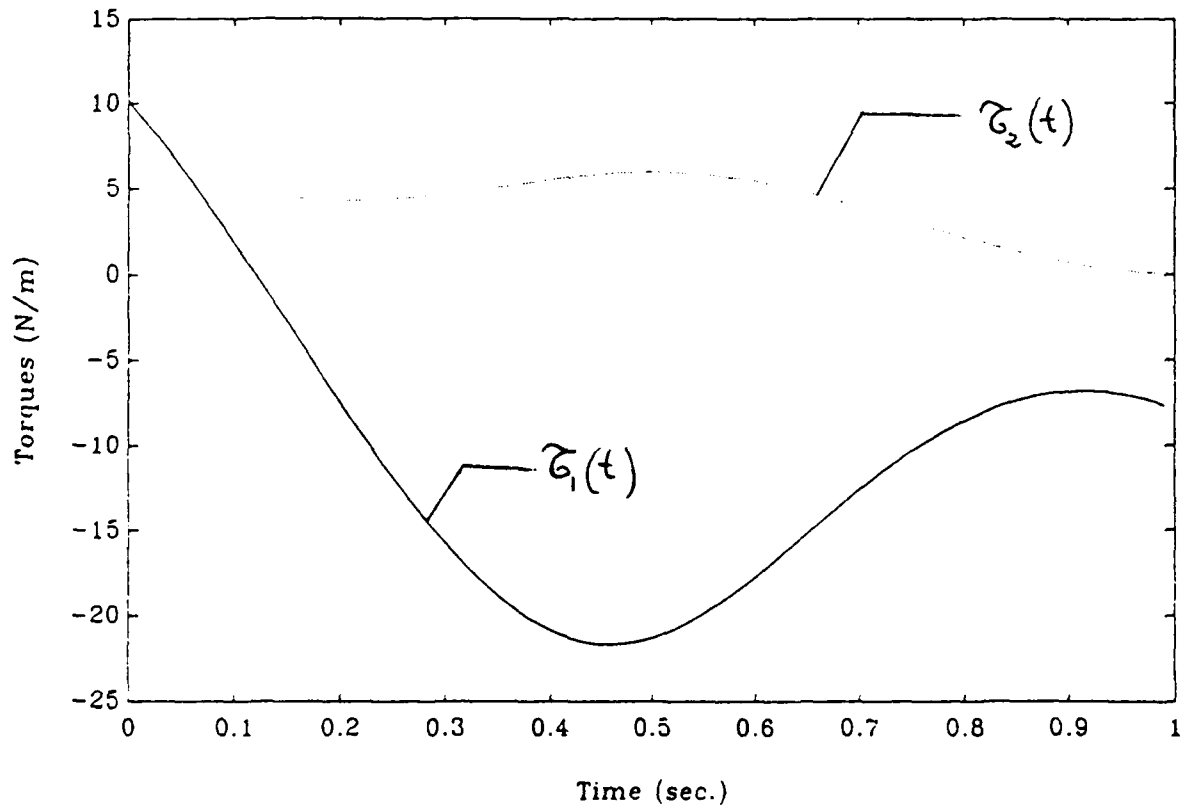


Figure 3.11: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Inverse Dynamics Controller (Approximate Modeling)

2. Variable Structure System

A different approach which yields a simple and robust control can be obtained by using a variable structure strategy. The theory of variable structure control has mostly been developed in the Soviet Union over the last 25 years and has found applications in many industrial processes [72]. The fundamental idea behind the theory of variable structure is to allow the controller to switch between different strategies, according to appropriate functions of the trajectory error.

A variable structure control is of the form:

$$u_{pi}(X_p(t), t) = \begin{cases} u_{pi}^+(X_p(t), t), & S_i(e_i) > 0 \\ u_{pi}^-(X_p(t), t), & S_i(e_i) < 0 \end{cases} \quad (3.24)$$

for $i = 1, 2, \dots, n$, where u_{pi} is the i^{th} component of the input vector U_p and, $S_i(e_i) = c_i e_i + \dot{e}_i$, $c_i > 0$, is the i^{th} component of the switching hypersurfaces. The design problem consists of choosing the functions u_{pi}^+ , u_{pi}^- , and the switching hyperplane matrix $C = \text{diag}(c_i)$ such that the sliding mode occurs on the switching hyperplanes, the tracking error has an acceptable transient response and it goes to zero asymptotically as $t \rightarrow \infty$. This methodology is simulated on the two link arm model. The desired trajectories are the same as before. The switching planes are chosen as:

$$S_1 = 0.5e_1 + \dot{e}_1$$

$$S_2 = 0.4e_2 + \dot{e}_2$$

In addition, to ensure the existence of the sliding modes, the control law is chosen as:

$$u_{pi}(X_p(t), t) = -[\alpha_i^1 |e_i| + \alpha_i^2 |\dot{e}_i| + \alpha_i^3 |e_2| + \alpha_i^4 |\dot{e}_2| + \alpha_i^5 |q_i^d| + \sigma_i] \text{sgn}(S_i)$$

In the absence of a procedural method to selecting the parameters α_i^j , a possible choice which will facilitate the calculations is:

$$\alpha_1^1 = 0, \alpha_1^2 = 0, \alpha_1^3 = 0, \alpha_1^4 = 0, \alpha_1^5 = 0 \text{ and } \sigma_1 = 10$$

$$\alpha_2^1 = 0, \alpha_2^2 = 0, \alpha_2^3 = 0, \alpha_2^4 = 0, \alpha_2^5 = 0 \text{ and } \sigma_2 = 20$$

The results of this computer simulation are reported in Figures 3.12 through 3.15. In Figure 3.12, the first link actual trajectory (q_1) tracks the desired trajectory (q_1^d) with approximately 7° error because of poor choice of the parameters a_i^j . The actual trajectory of the second link, however, shows better tracking as seen in Figure 3.13. Figure 3.14 shows the time evolution of the errors e_1 and e_2 in both joints. As expected, Figure 3.15 shows considerable chattering in the input signals. These simulation results highlight the fact that chattering in the input signals, absence of a procedural method to choosing the control parameters a_i^j , and the difficulty of guaranteeing the existence of the sliding modes are the main reasons that limit the applicability of this scheme to multivariable control systems.

3. Adaptive Linear Model Following Control

This scheme is depicted in Figure 3.16. The reference model is chosen to be a stable, linear, time invariant and decoupled system as:

$$\dot{X}_m(t) = A_m X_m(t) + B_m U_m(t) \quad (3.25)$$

where the torques $U_m(t)$ are selected so that the output $X_m(t)$ of the model follows precisely the desired trajectories, described by the user. A_m and B_m are of the form:

$$A_m = \begin{bmatrix} 0 & \vdots & I_n \\ \dots & \dots & \dots \\ -\text{diag}(a_{m0i}) & \vdots & -\text{diag}(a_{m1i}) \end{bmatrix} \quad (3.26)$$

$$B_m = \begin{bmatrix} 0 \\ \dots \\ I_n \end{bmatrix} \quad (3.27)$$

with

$$a_{m0i} > 0, a_{m1i} > 0$$

For the purpose of simulation, $a_{m0i} = 1.5$ and $a_{m1i} = 2.5$, $i = 1, 2$. The manipulator input U_p is chosen as:

$$U_p(v, X_p, t) = \Phi(v, X_p, t) X_p - K_p X_p + \Psi(v, X_p, t) U_m + K_u U_m \quad (3.28)$$

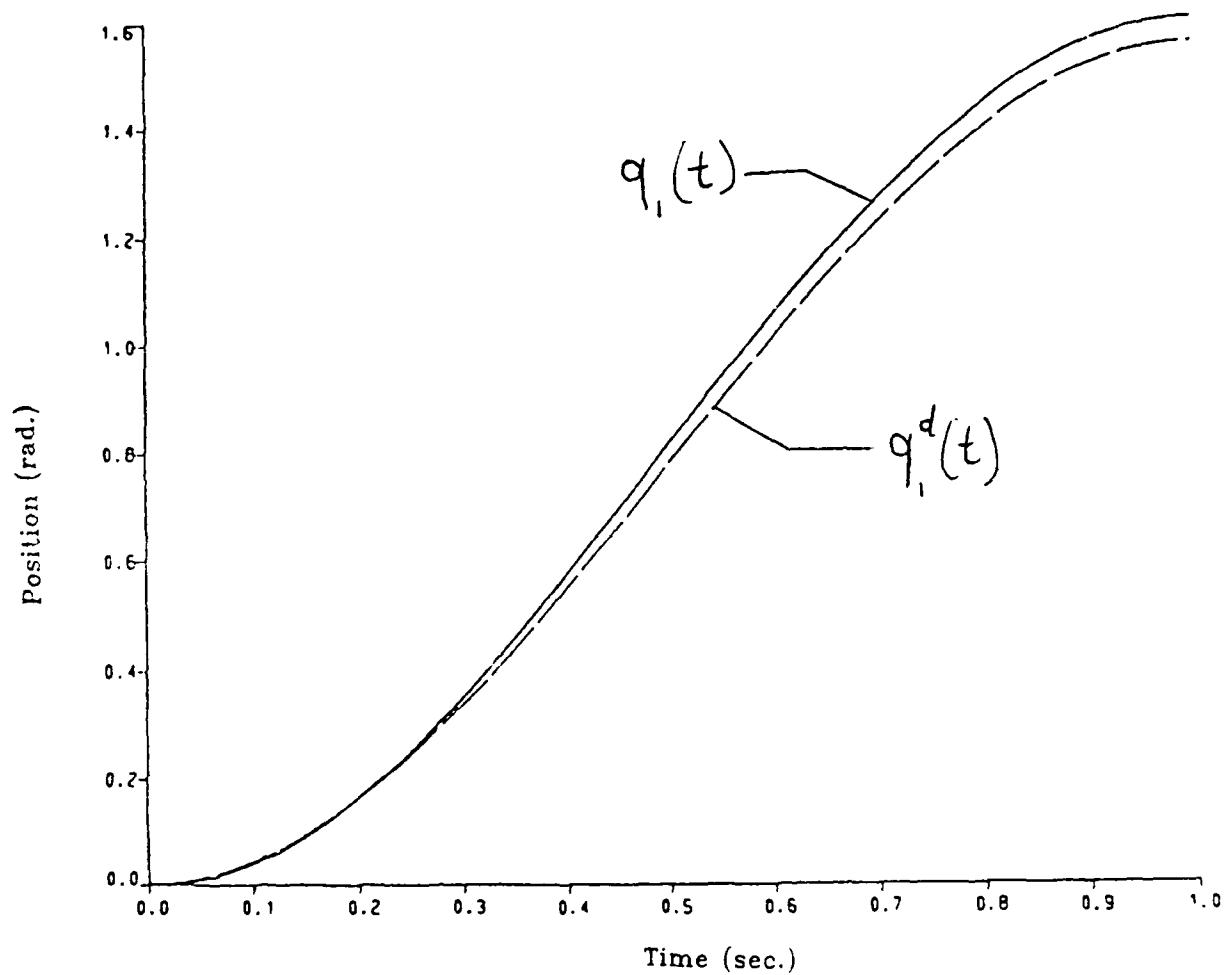


Figure 3.12: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Variable Structure Controller (Perfect Modeling)

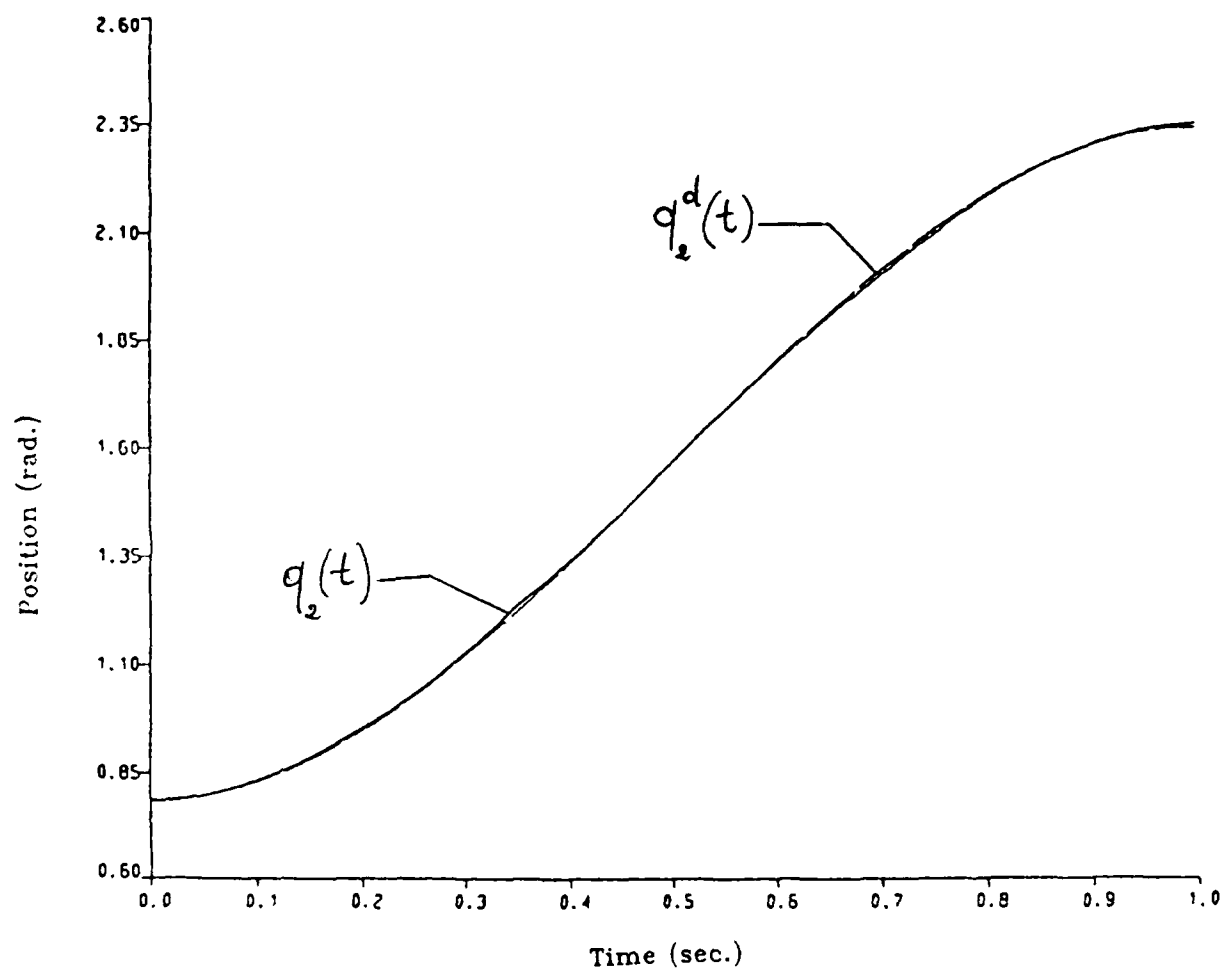


Figure 3.13: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Variable Structure Controller (Perfect Modeling)

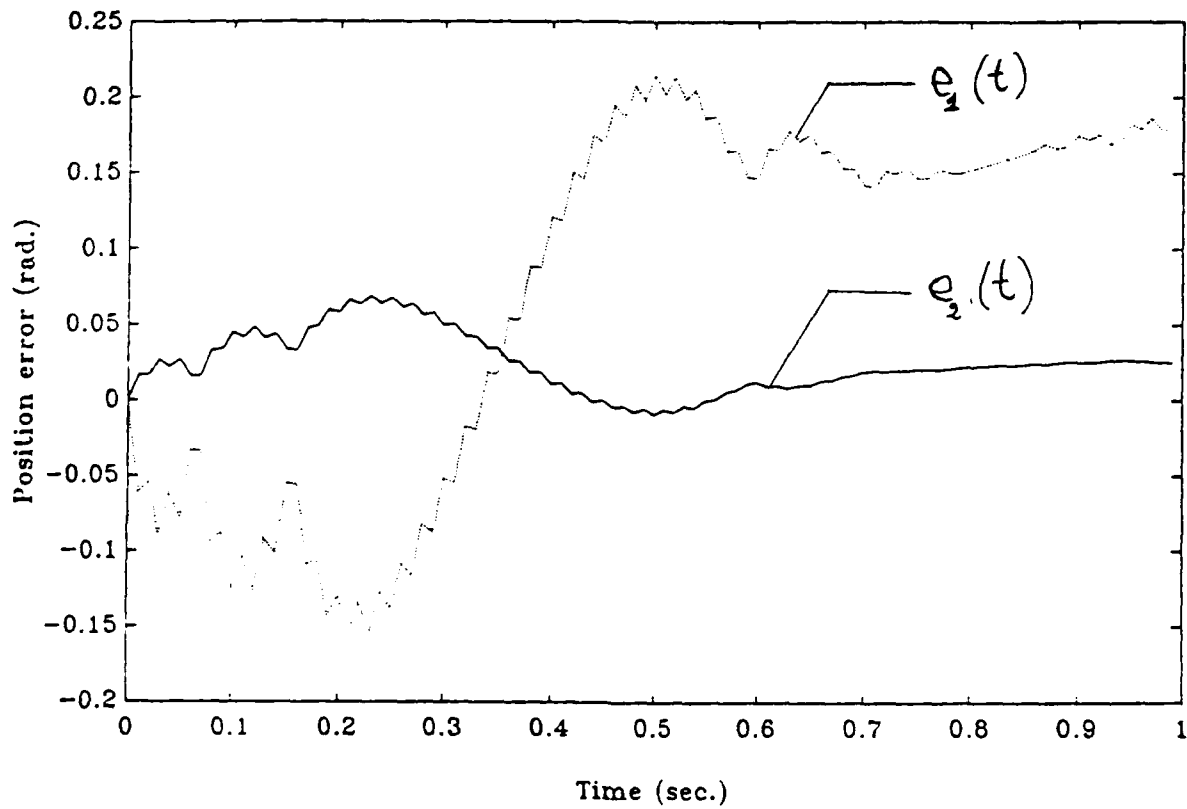


Figure 3.14: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Variable Structure Controller (Perfect Modeling)

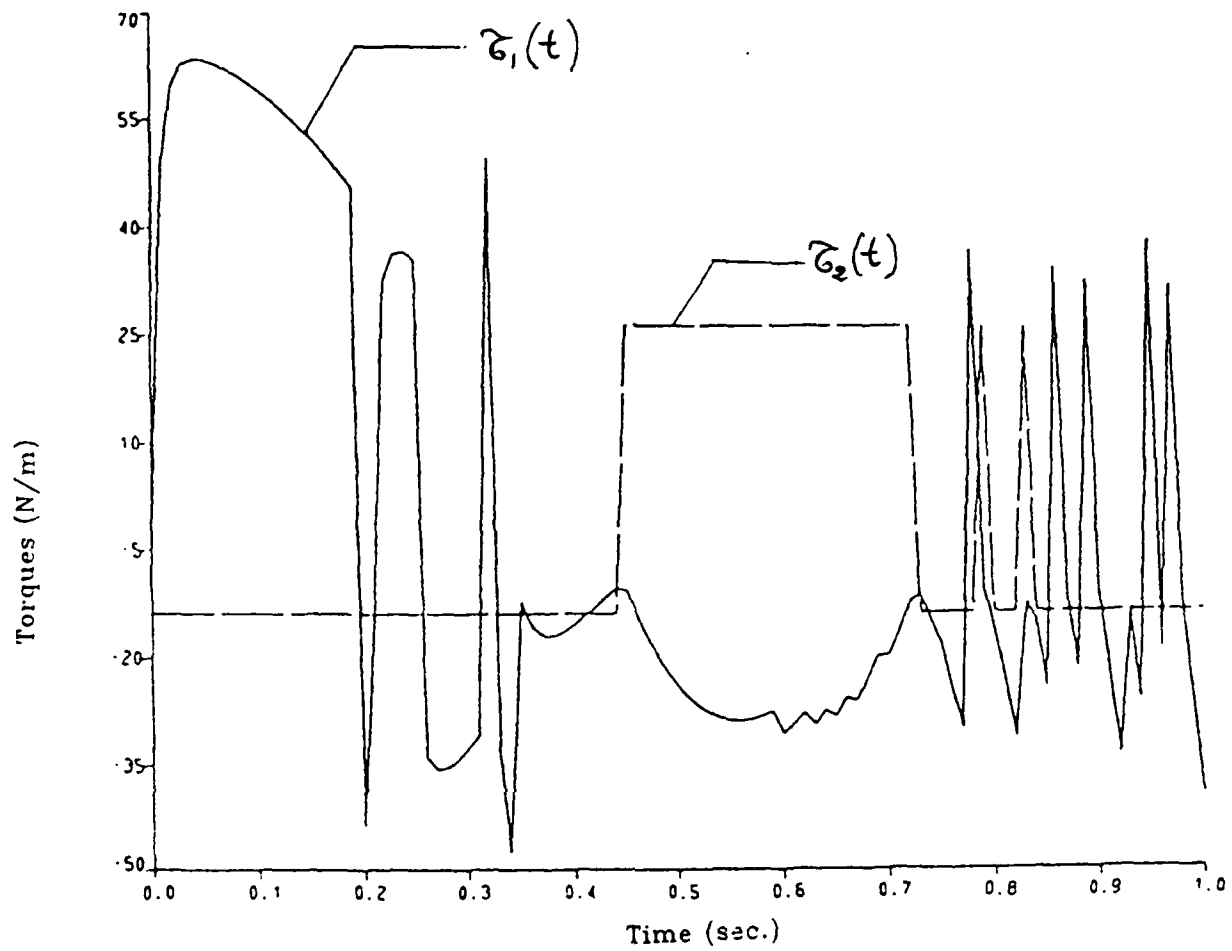


Figure 3.15: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Variable Structure Controller (Perfect Modeling)

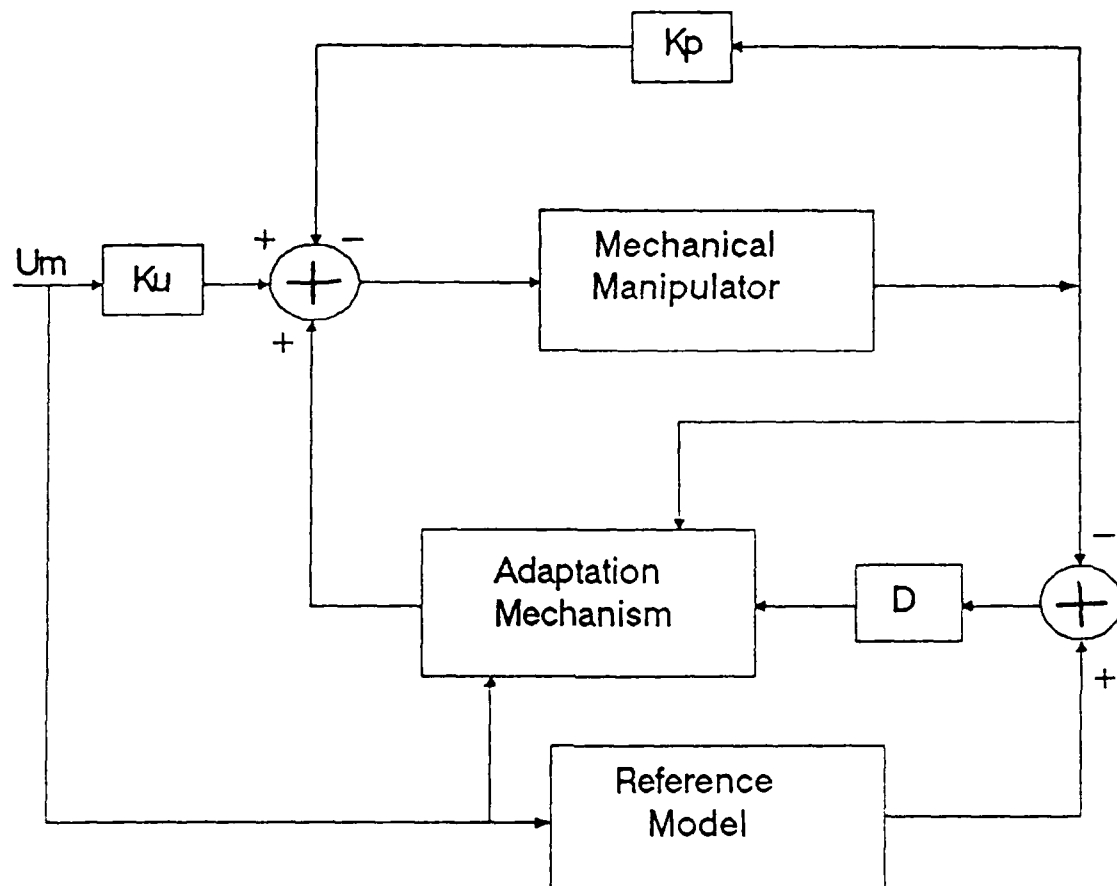


Figure 3.16: A Linear Adaptive Model Following Control for mechanical manipulators

where K_p and K_u are feedback constant gain matrices designed for specific nominal values of the plant to satisfy the perfect model following conditions given by:

$$K_p = -A^+(q(t)) (A_m - A_p) \quad (3.29)$$

$$K_u = A^+(q(t)) B_m \quad (3.30)$$

with $A^+(q(t))$ being the pseudoinverse of $A(q(t))$.

The quantities Φ and Ψ are generated by the adaptation mechanism to guarantee the stability of the overall system. Possible choices are:

$$\Phi = \zeta \frac{v}{\|v\|} (\text{sgn}(X_p))^T \quad (3.31)$$

$$\Psi = \xi \frac{v}{\|v\|} (\text{sgn}(U_m))^T \quad (3.32)$$

with

$$\zeta \geq \frac{[\lambda_{\max}(RR^T)]^{.5}}{\lambda_{\min}(A^{-1}(q(t)))} \quad (3.33)$$

$$\xi \geq \frac{[\lambda_{\max}(SS^T)]^{.5}}{\lambda_{\min}(A^{-1}(q(t)))} \quad (3.34)$$

where

$$R = A^{-1}(q(t))A^+(q(t))(A_m - A_p) + A^{-1}(q(t))K_p \quad (3.35)$$

$$S = A^{-1}(q(t))A^+(q(t))B_m - A^{-1}(q(t))K_u \quad (3.36)$$

Simulation results of this technique with K_p and K_u obtained from equations (3.29) and (3.30) at $t = 0$ as:

$$K_p = K_p(0) = \begin{bmatrix} -.354 & 30.4 & 6.77 & 1.72 \\ 30.4 & 4.91 & 1.72 & .833 \end{bmatrix}$$

$$K_u = K_u(0) = \begin{bmatrix} 2.71 & .687 \\ .687 & .333 \end{bmatrix}$$

are reported in Figures 3.17 through 3.20. Since the matching matrices K_p and K_u are not adjusted in time, we can see that the actual trajectories (q_1 and q_2) follow the desired

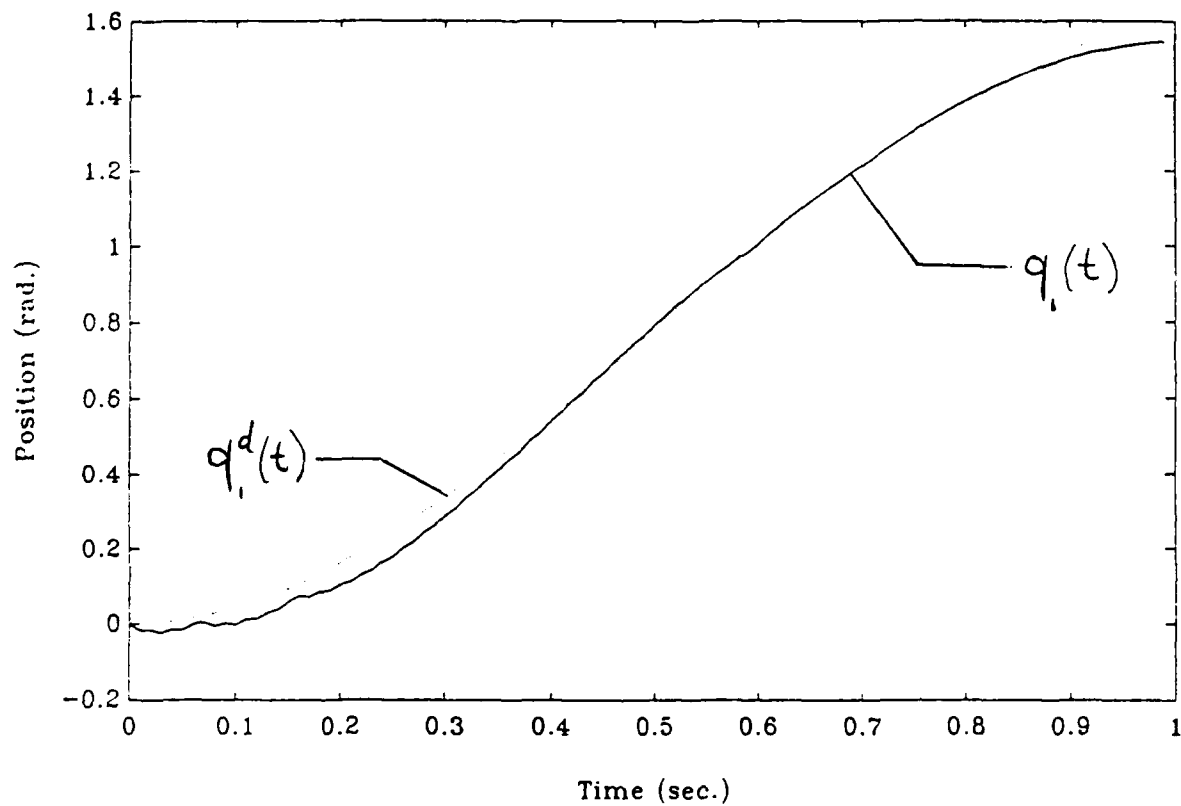


Figure 3.17: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Adaptive Model Following Controller (Matching Matrices not adjusted)

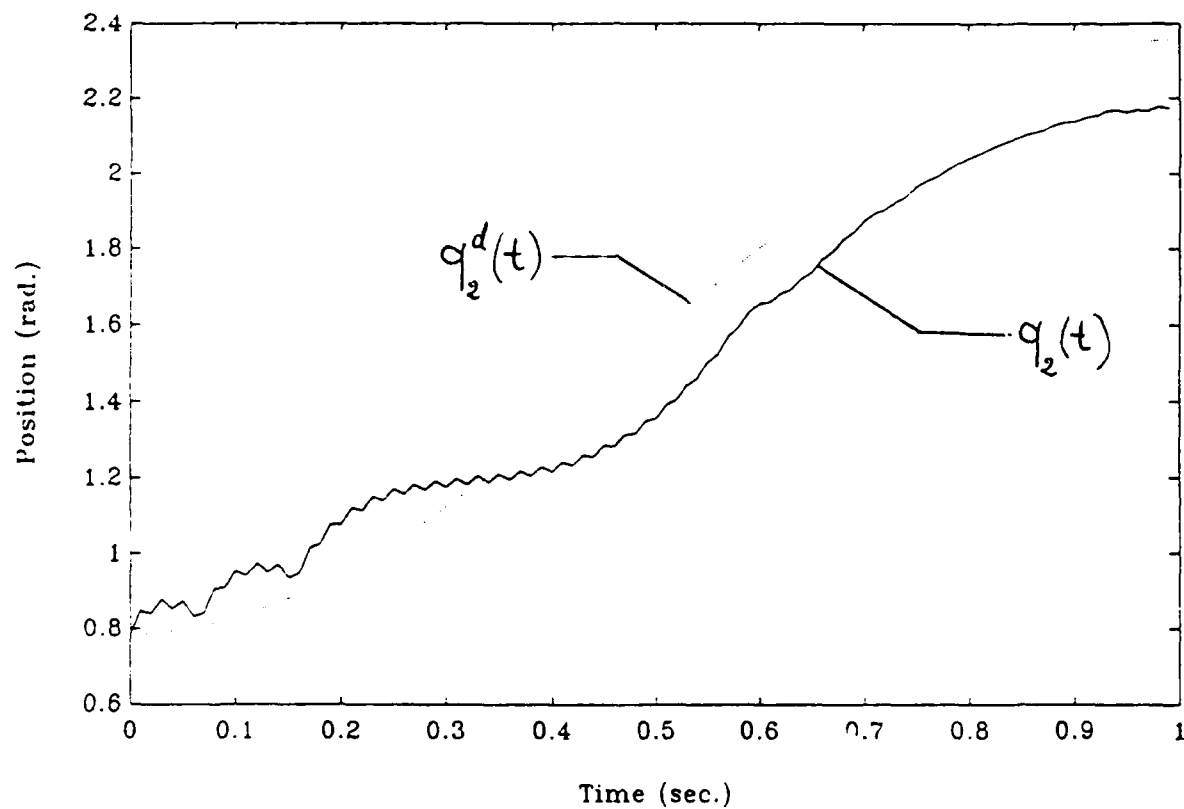


Figure 3.18: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Model Following Controller (Matching Matrices not adjusted)

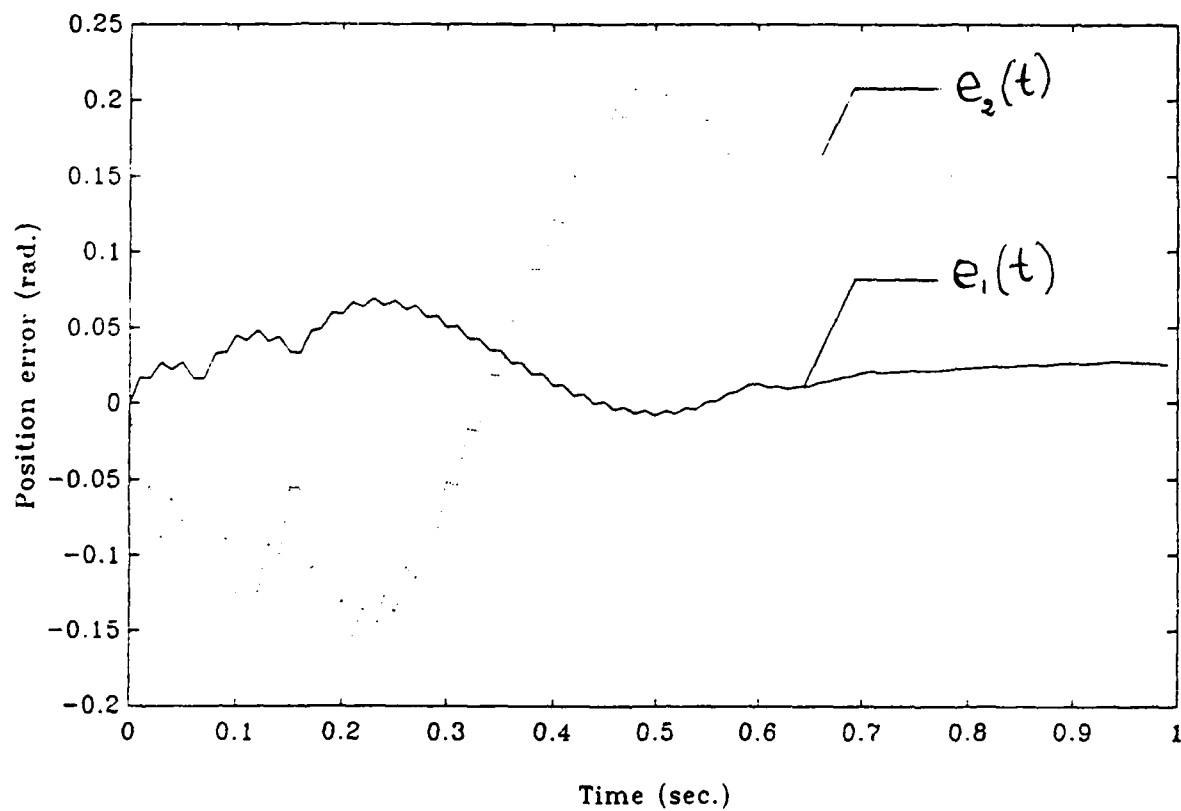


Figure 3.19: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Model Following Controller (Matching Matrices not adjusted)

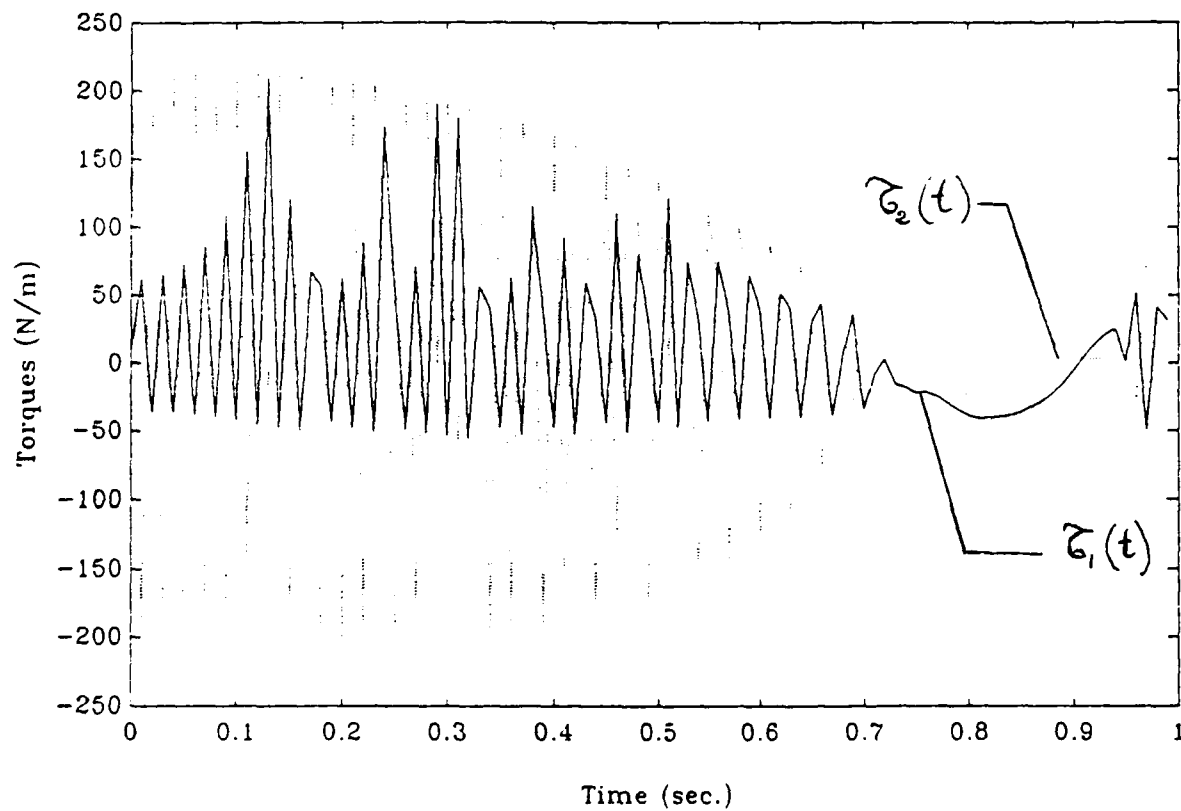


Figure 3.20: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller (Matching Matrices not adjusted)

trajectories (q_1^d and q_2^d) with an error of 4° in the first link (Figure 3.17) and of 7° in the second link (Figure 3.18). The joints errors (e_1 and e_2) are shown in Figure 3.19. The control signals (Figure 3.20) are chattering due to the high frequency component.

Using this type of adaptation, on line numerical integration of the dynamics equations of motion is avoided. However, the signals that the actuators are required to generate (Figure 3.20) are about 10 times larger than in the computed torque (Figure 3.11). This is a serious threat to the plant hardware since the forcing signals are discontinuous.

To be able to reduce the parameters ζ and ξ , and hence, the actuation signals, one should calculate K_p and K_u that will satisfy the perfect model following conditions of equations (3.29) and (3.30) at each instant of time. Simulation results obtained using this fact are given in Figures 3.21 through 3.24. In this case, the trajectory of the first link (Figure 3.21) as well as the trajectory of the second link (Figure 3.22) show very close tracking. the position errors in both the first and the second links are reduced to zero (Figure 3.23). The actuation signals (Figure 3.24) are still large. The main disadvantage of this choice is, however, the added computational complexity.

4. Adaptive Perturbation Control

A block diagram of this scheme is shown in Figure 3.25. This methodology uses an available nominal model of the system and the recursive Newton–Euler equations of motion to compute nominal control inputs for a given trajectory. These nominal torques compensate for all the interaction forces among various joints along the nominal trajectory.

To compensate for small deviations from the nominal trajectory, a feedback adaptive component is introduced. This adaptive control is based on linearizing the manipulator dynamics equations in the vicinity of known nominal trajectory set points to obtain the associated perturbed state equation:

$$\dot{e} = Ae + Bdr \quad (3.37)$$

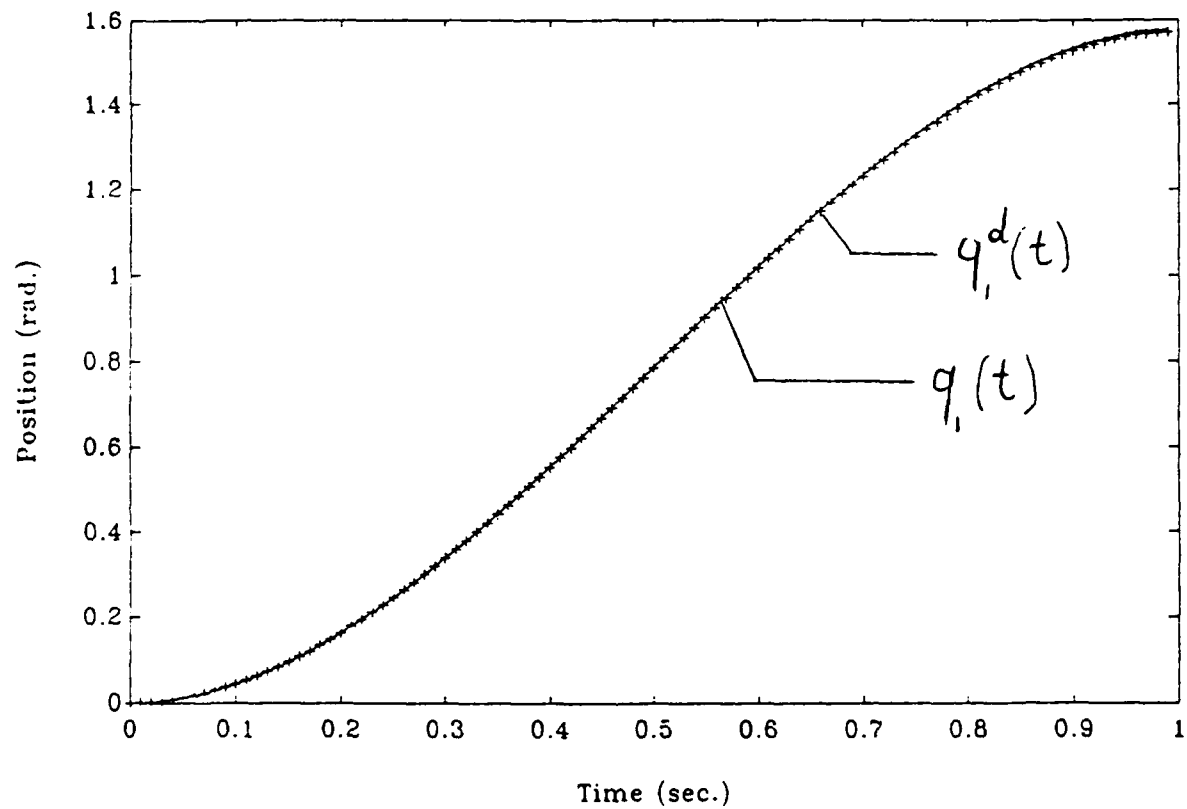


Figure 3.21: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Adaptive Model Following Controller (Matching Matrices adjusted)

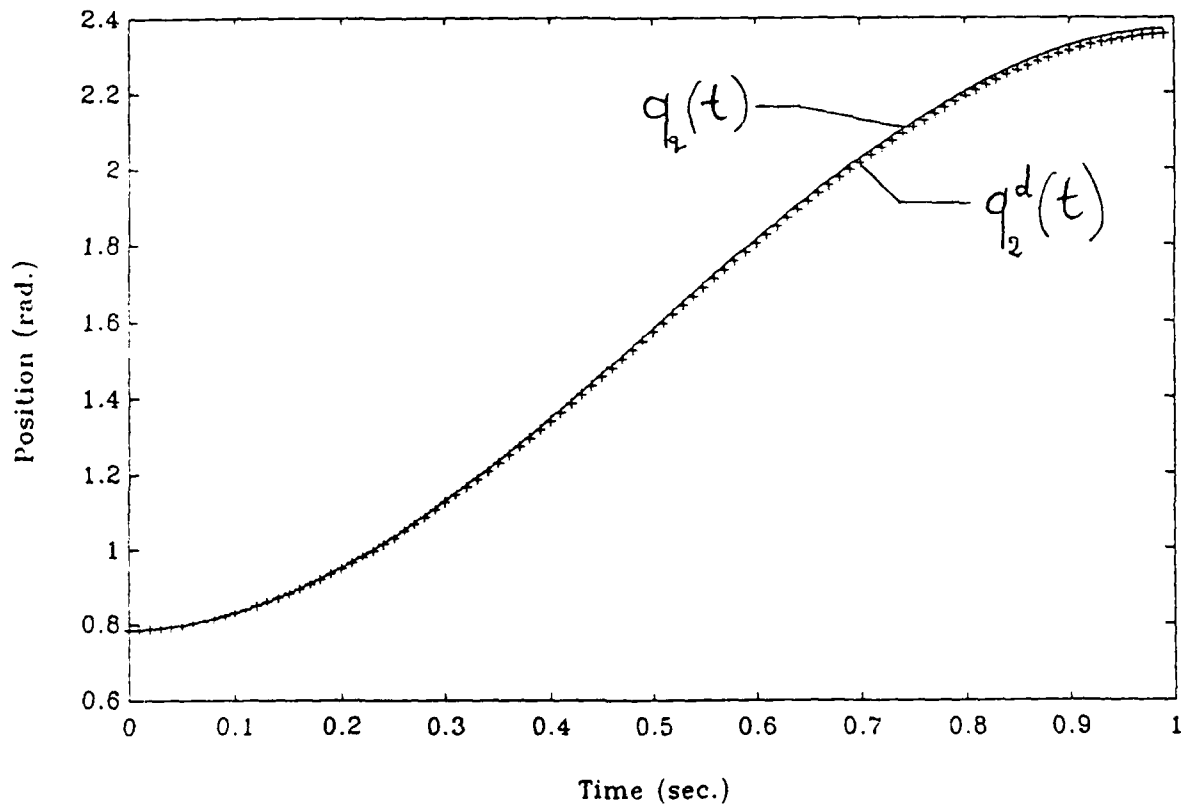


Figure 3.22: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Model Following Controller (Matching Matrices adjusted)

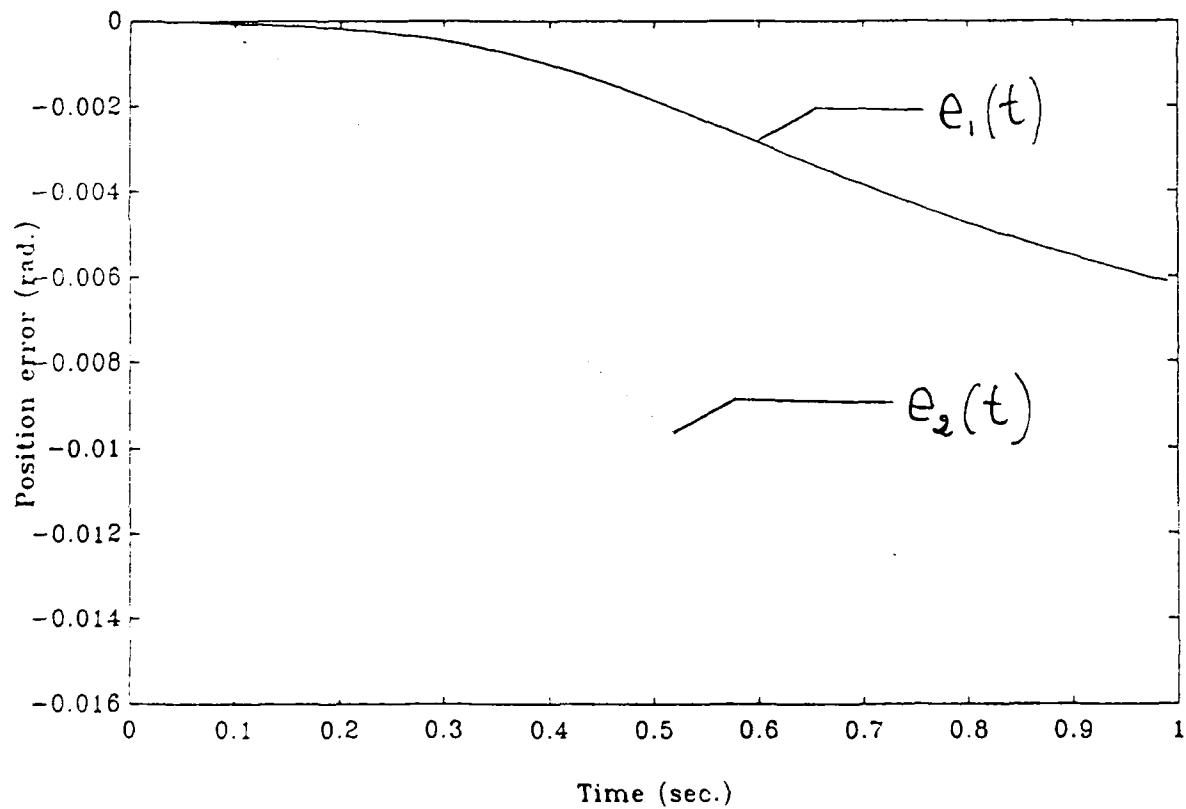


Figure 3.23: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Model Following Controller (Matching Matrices adjusted)

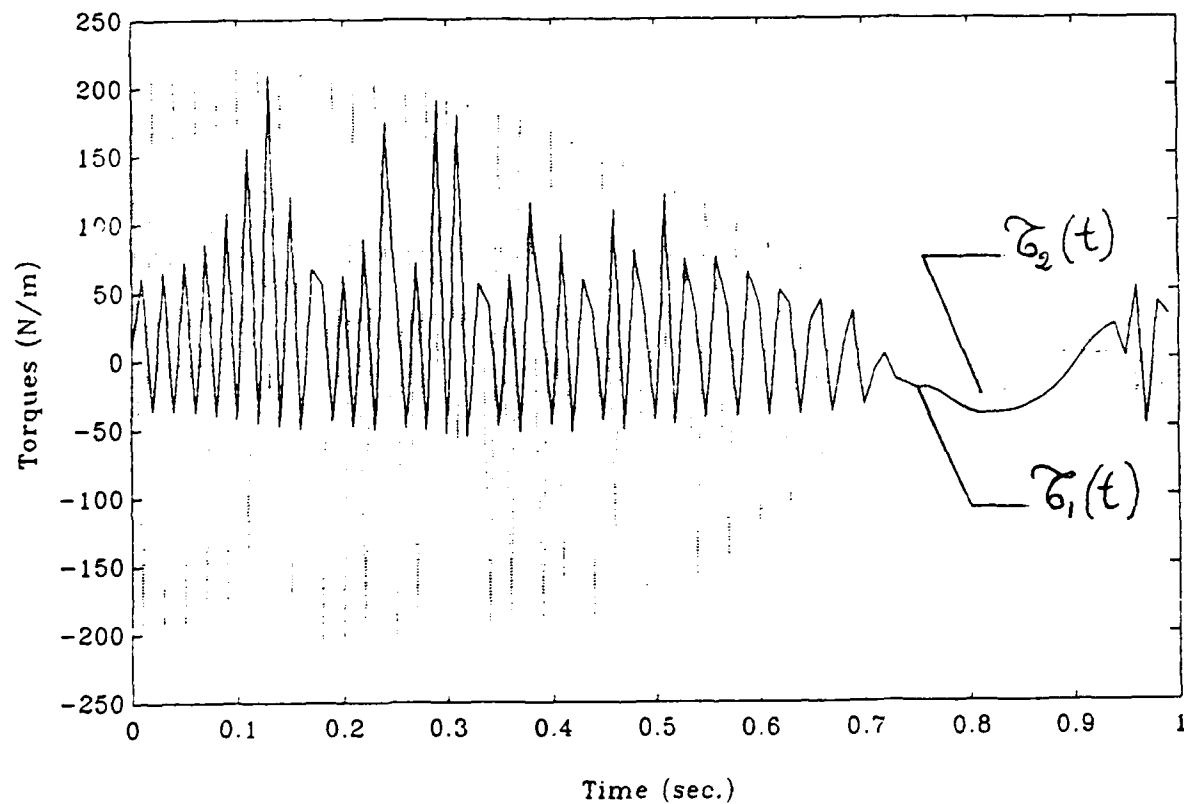


Figure 3.24: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller (Matching Marices adjusted)

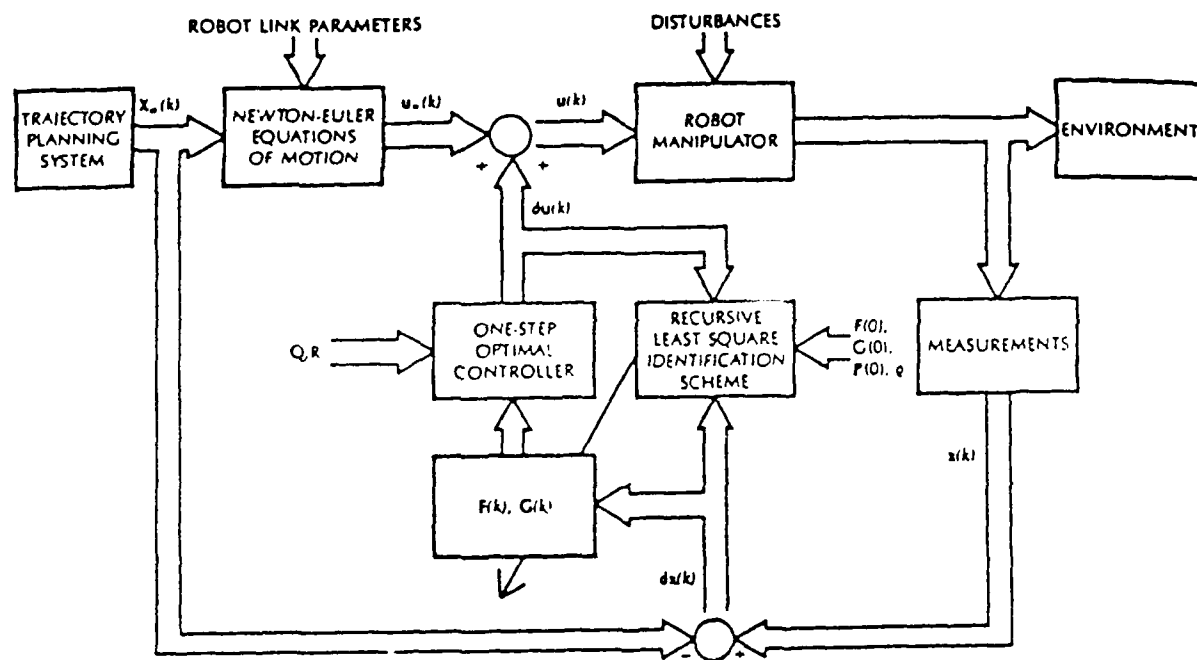


Figure 3.25: Adaptive Perturbation Control

where e and \dot{e} are as defined earlier and $d\tau = U - U_n$, U_n being the nominal torque inputs as obtained from certain available nominal model of the manipulator. The system parameters A and B depend on the instantaneous manipulator position and velocity along the nominal trajectory. A recursive least squares parameters identification technique is used to estimate the unknown elements in A and B . The obtained parameters are then used to formulate a one step optimal controller that will generate the torques $d\tau$ to compensate for the perturbations. When only the feedforward torques are implemented, simulation results show that 10% error in the load produces tracking errors in both the first (Figure 3.26) and the second (Figure 3.27) links. As seen in Figure 3.28, these tracking errors are in the order of 6° and of 9° , respectively. The input torques (Figure 3.29) stay within reasonable limits. Figures 3.30 through 3.33 give the results of the same simulations as above when both the feedforward and the correcting torques are used. While an improved tracking is experienced in both the first (Figure 3.30) and the second (Figure 3.31) links, the joint errors (Figure 3.32) are still of the order of 5° and 3° , respectively. The input control signals (shown in Figure 3.33) stay within the same range of values as before. These results are, however, expected since this strategy assumes slow variations and small deviations about the desired trajectory.

It is evident from the above discussion that in order to extend the capabilities of manipulators and improve their overall dynamic performances, there is a need to investigate and develop better adaptive control solutions to current control problems.

The aim of the next section is to present a novel adaptive control law for mechanical manipulators that enjoys global stability and overcomes some of the limitations of the previously studied methodologies. This strategy combines properties from both the the Self Tuning Regulator in [51] and the Model Reference Adaptive Control in [63] and offers itself to microcomputer implementation. This technique serves also to extend the Model Reference Adaptive Control method into using a nonlinear reference model.

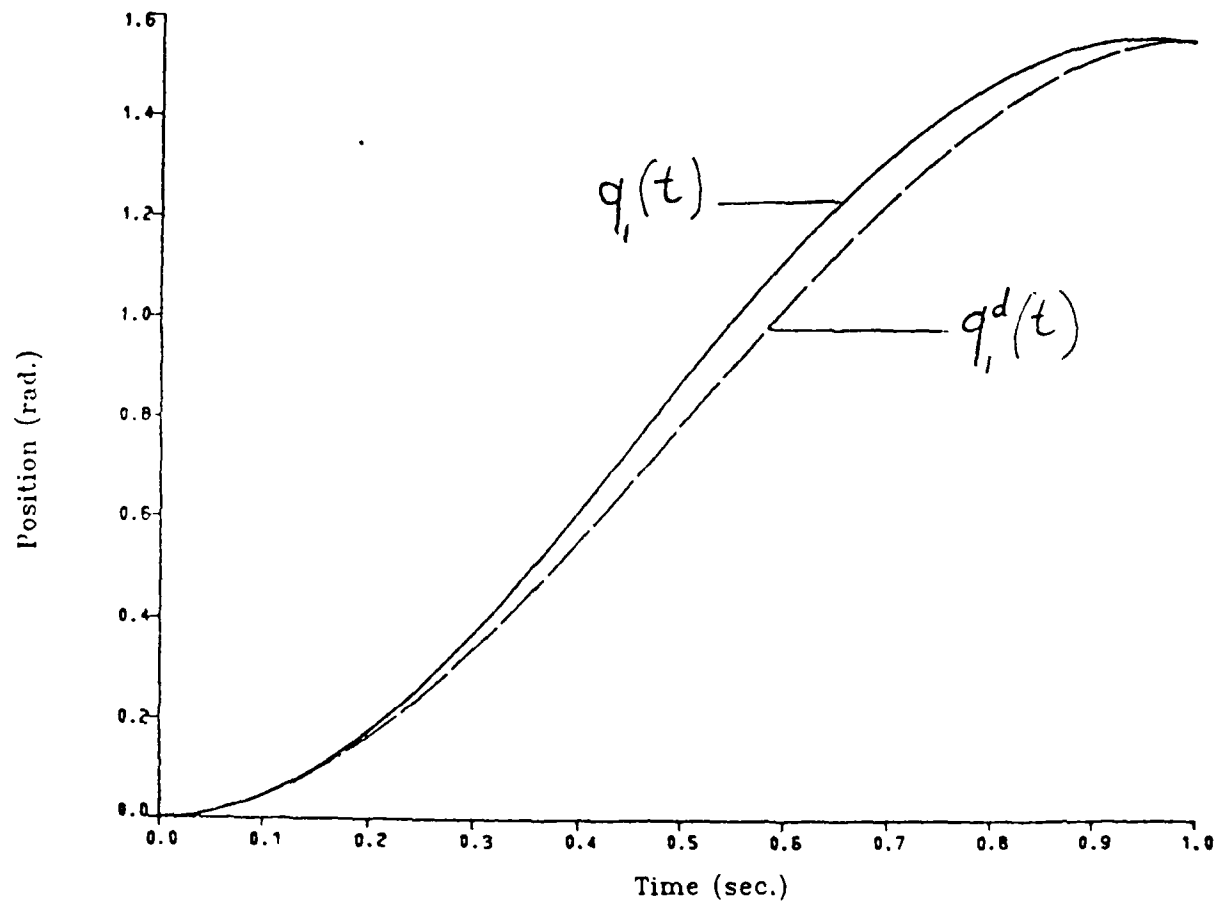


Figure 3.26: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Adaptive Perturbation Controller (No Adaptation)

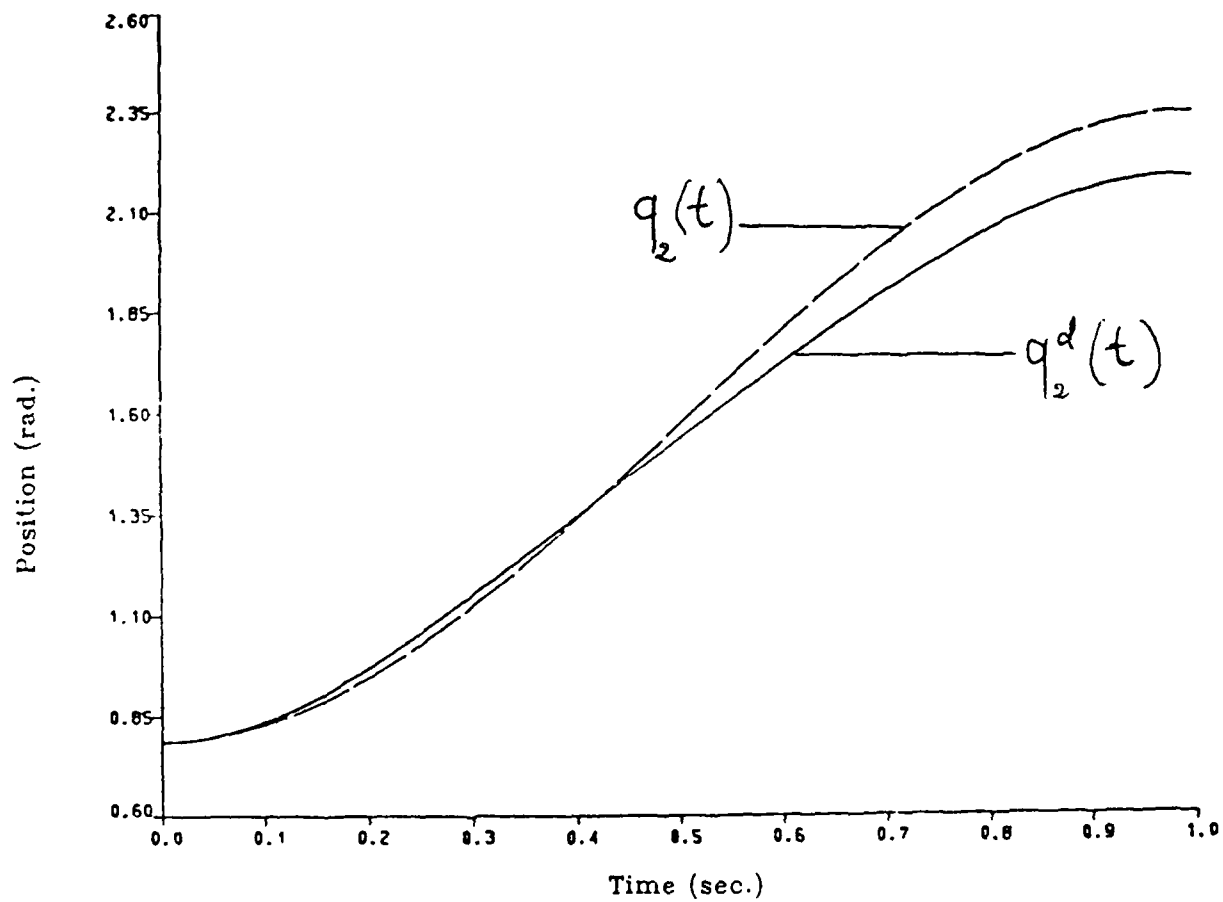


Figure 3.27: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Perturbation Controller (No Adaptation)

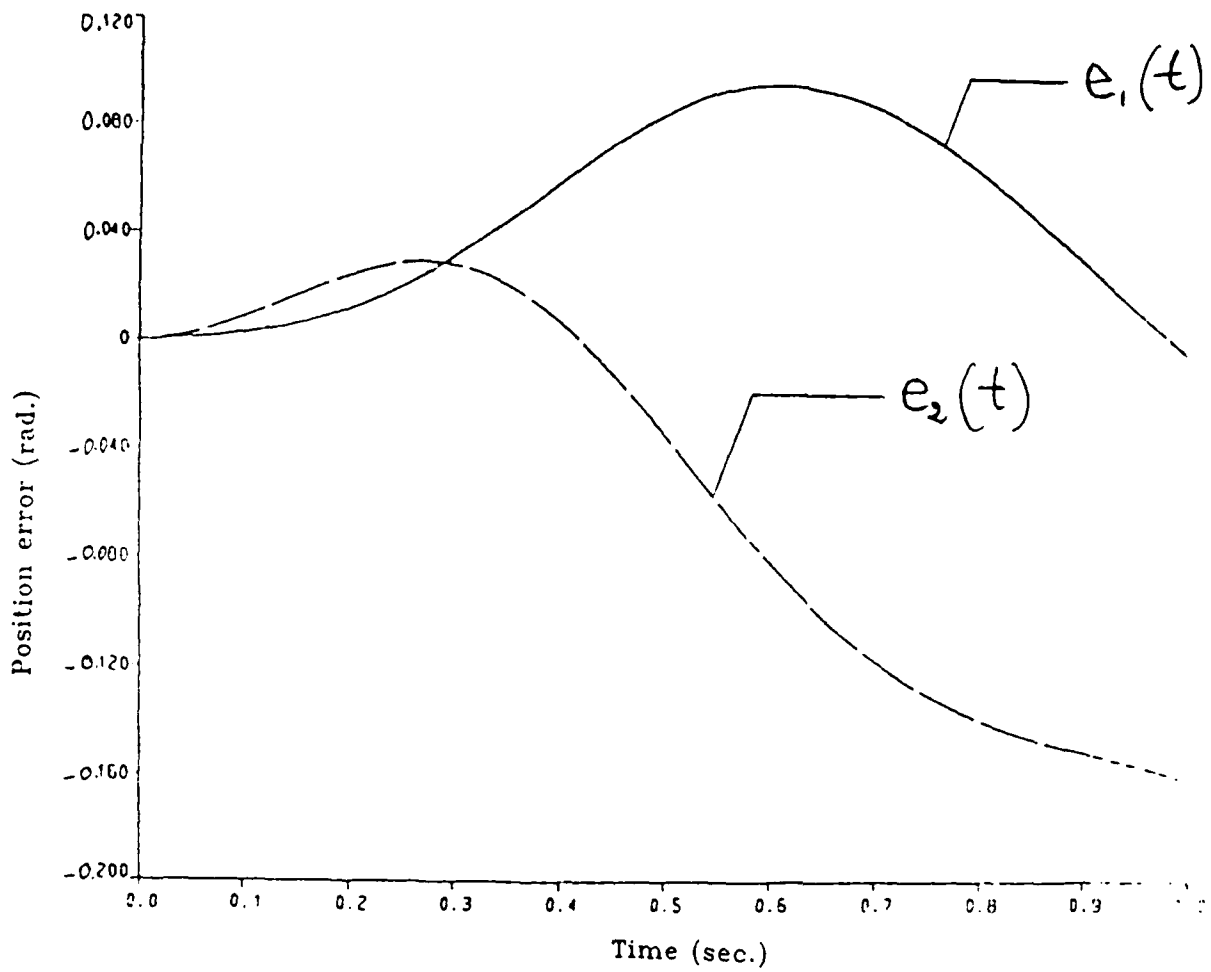


Figure 3.28: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Perturbation Controller (No Adaptation)

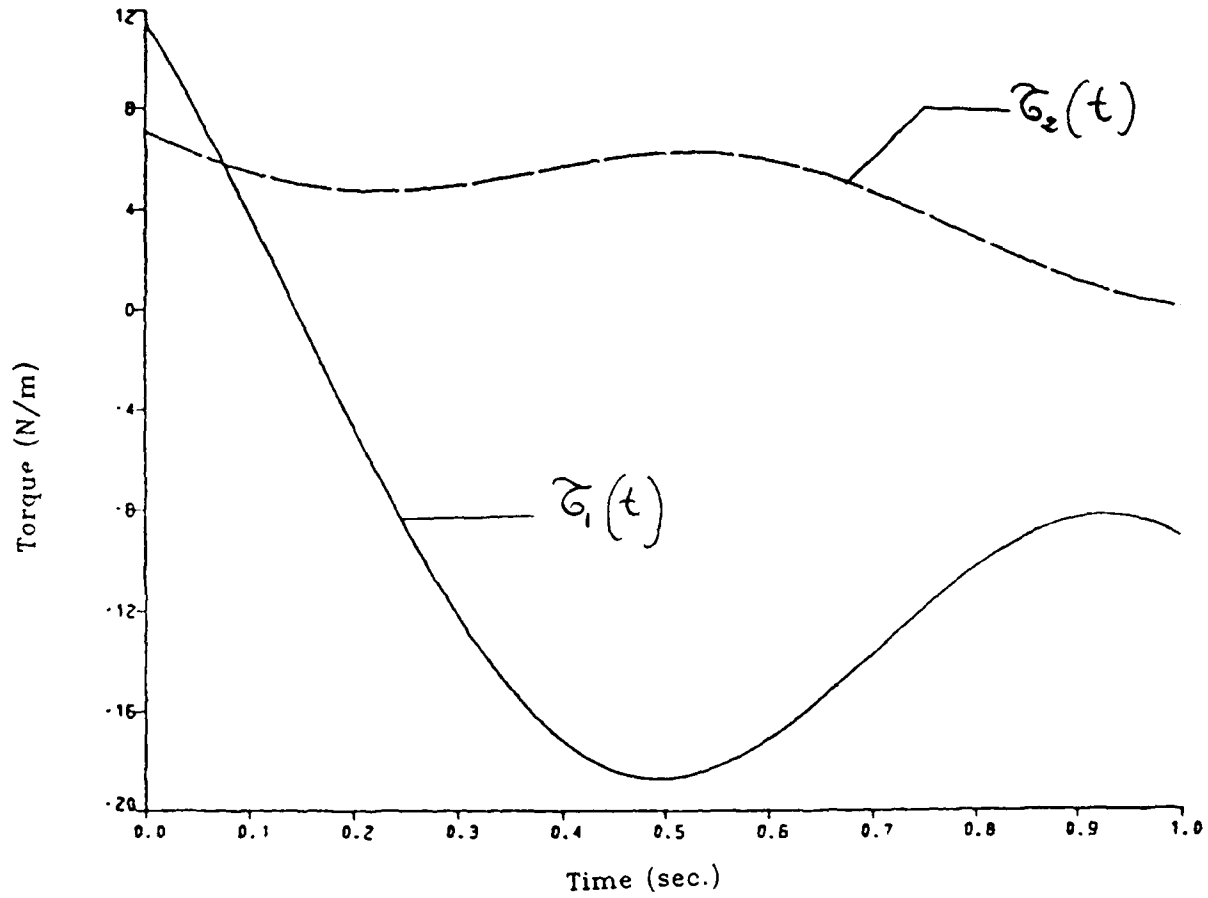


Figure 3.29: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Perturbation Controller (No Adaptation)

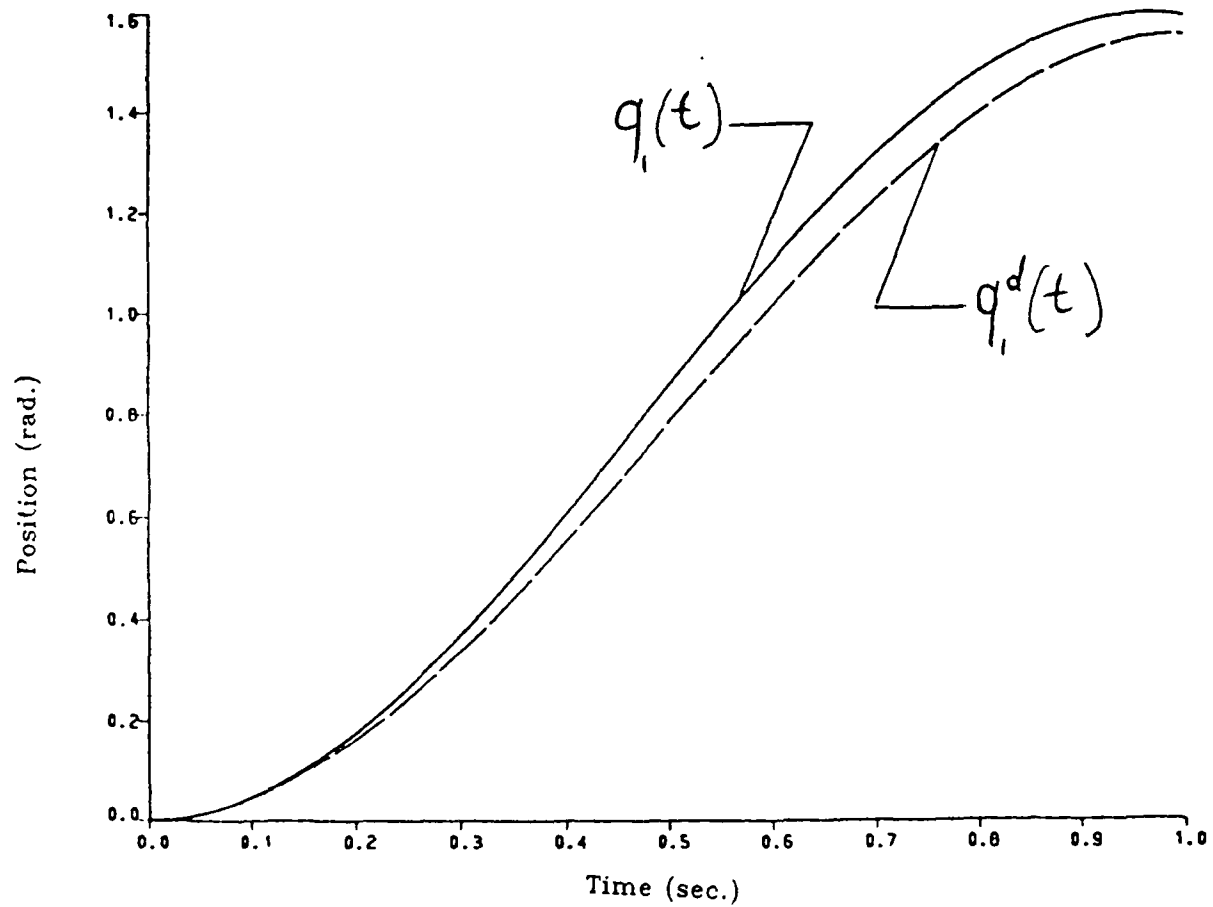


Figure 3.30: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Adaptive Perturbation Controller (With Adaptation)

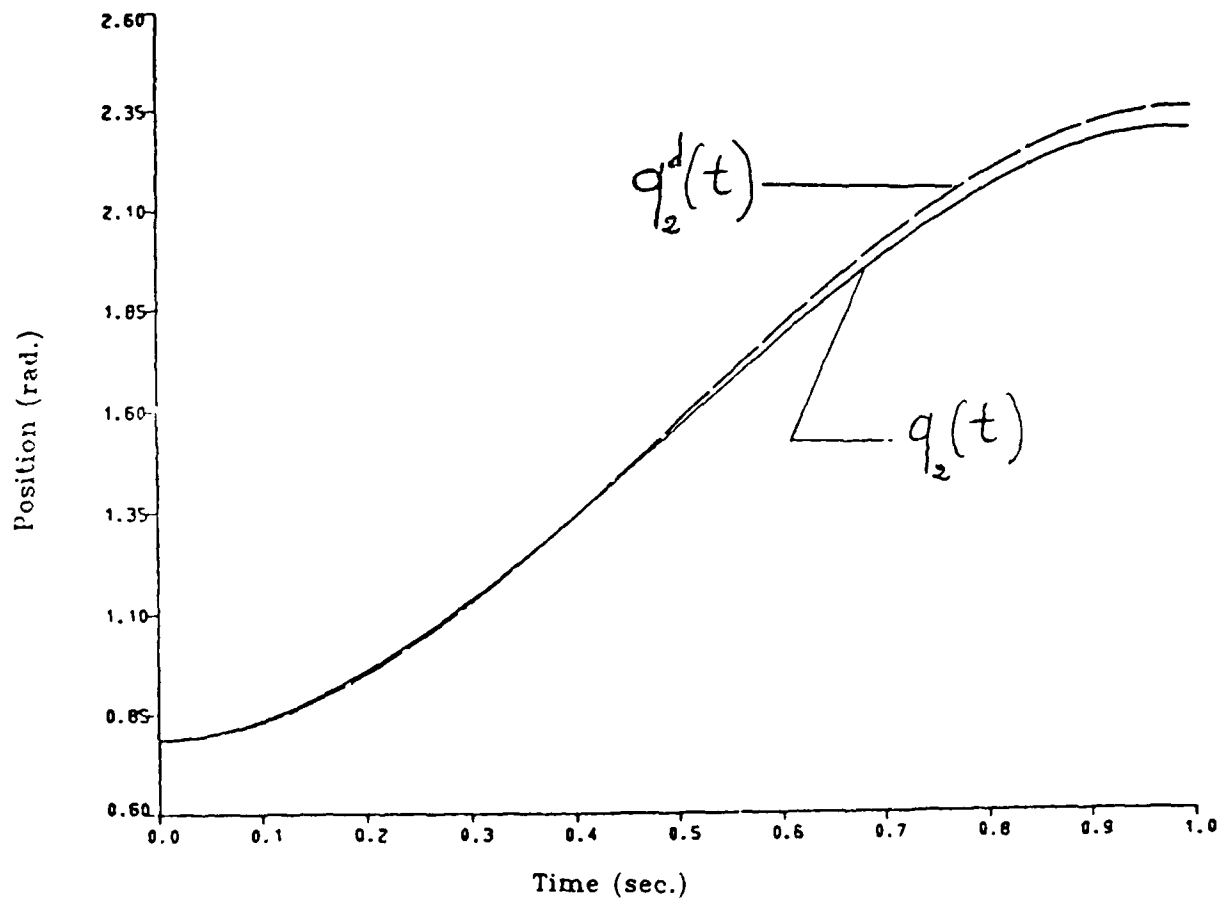


Figure 3.31: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Perturbation Controller (With Adaptation)

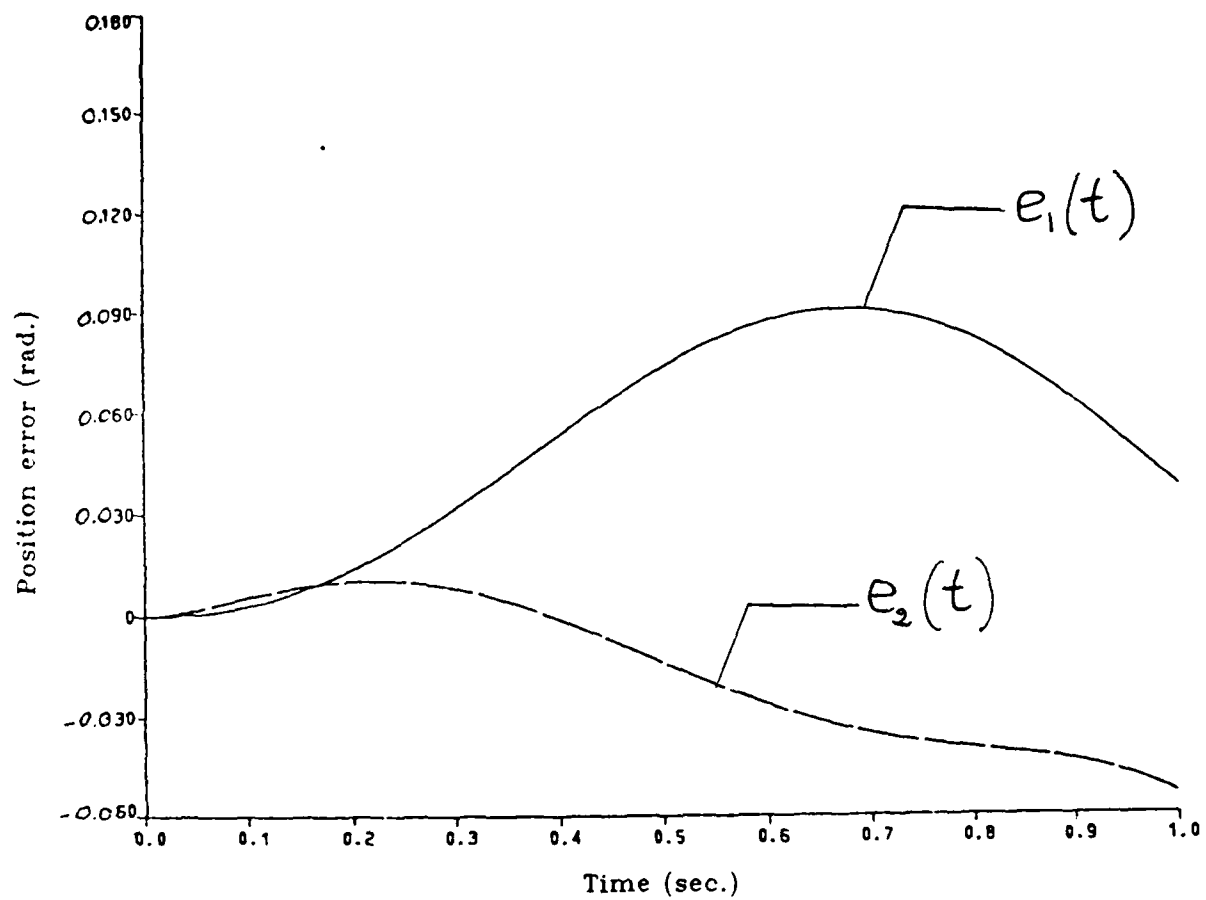


Figure 3.32: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Perturbation Controller (With Adaptation)

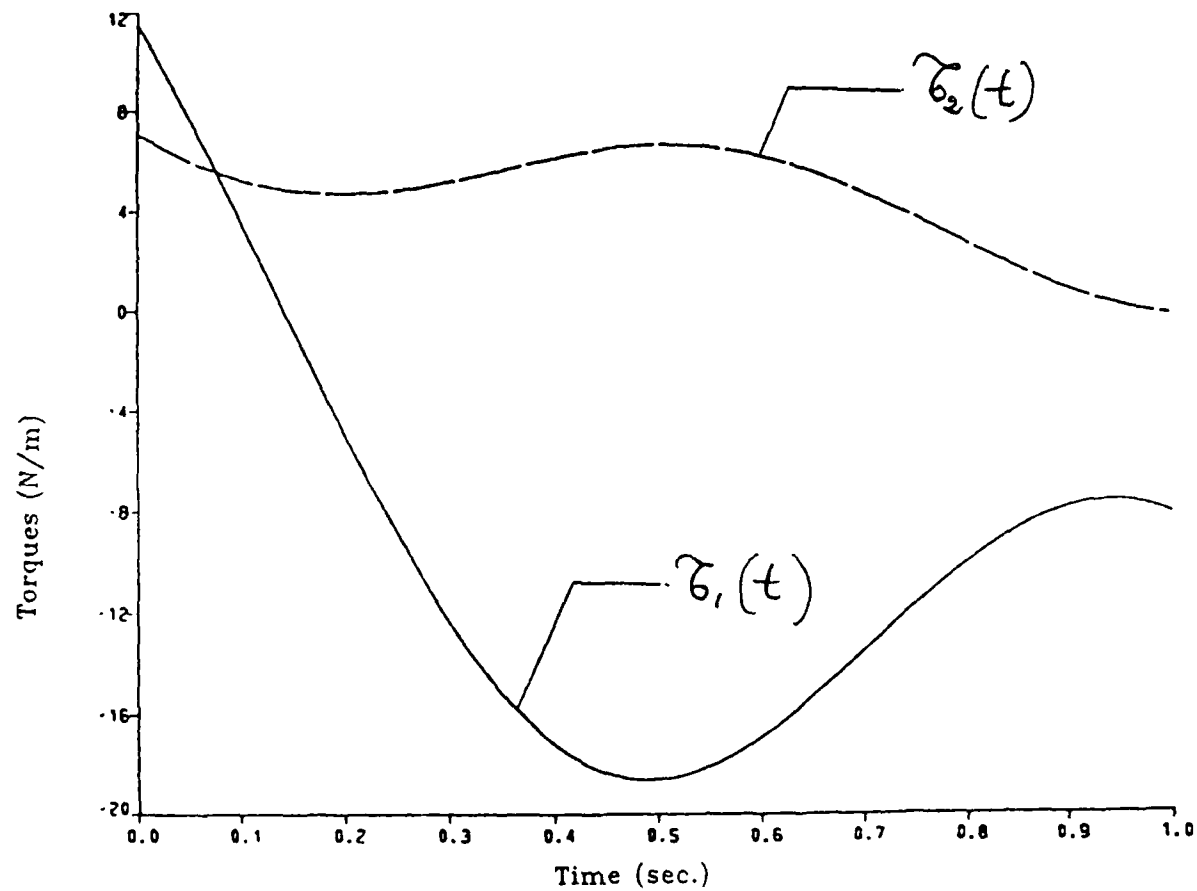


Figure 3.33: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Perturbation Controller (With Adaptation)

D. ADAPTIVE NONLINEAR MODEL FOLLOWING CONTROL

Figure 3.34 illustrates the structure of the proposed adaptive control system. The task of the controller is to generate the control signals in order to follow a desired trajectory despite the changes in the manipulator's parameters and the errors in the dynamic model. The desired trajectory is specified in terms of joint angles $q^d(t)$ and their derivatives $\dot{q}^d(t)$ and $\ddot{q}^d(t)$. The total torques τ are obtained through a nominal and a correction loops.

The nominal loop is justified by the fact that, in practice, a nominal model of the manipulator is always available to the designer. The nominal parameters are used to construct a recursive inverse dynamics that generates nominal torques τ_n . The inputs q_n , \dot{q}_n , and \ddot{q}_n to the recursion are obtained by adding filtered error signals $z(t)$, $\dot{z}(t)$, and $\ddot{z}(t)$ to the desired signals q^d , \dot{q}^d , and \ddot{q}^d . The signals $z(t)$ are chosen such that:

$$Z(s) = F(s)E(s) \quad (3.38)$$

where $F(s)$ is the transfer function of a linear filter to be determined and $Z(s)$ and $E(s)$ are Laplace transform of $z(t)$ and $e(t)$ respectively. The recursive form is chosen to ease the computation. The inverse dynamics form is motivated by the fact that, in the case where the mechanical manipulator model is known precisely, an exact trajectory following is obtained. The adjusting signals $z(t)$ are selected to reach the ideal closed loop dynamics given by the error equation:

$$\ddot{e}(t) + K_v \dot{e}(t) + K_p e(t) = 0 \quad (3.39)$$

Since some manipulator parameters such as load and inertia vary in time and some others such as friction in the gears and motors backlash are very difficult to determine, additional correction feedback torques $d\tau$ are necessary to account for any deviations from the desired trajectory due to these effects. These correcting torques are generated to guarantee global stability:

$$\lim_{t \rightarrow \infty} e(t) = 0 \quad (3.40)$$

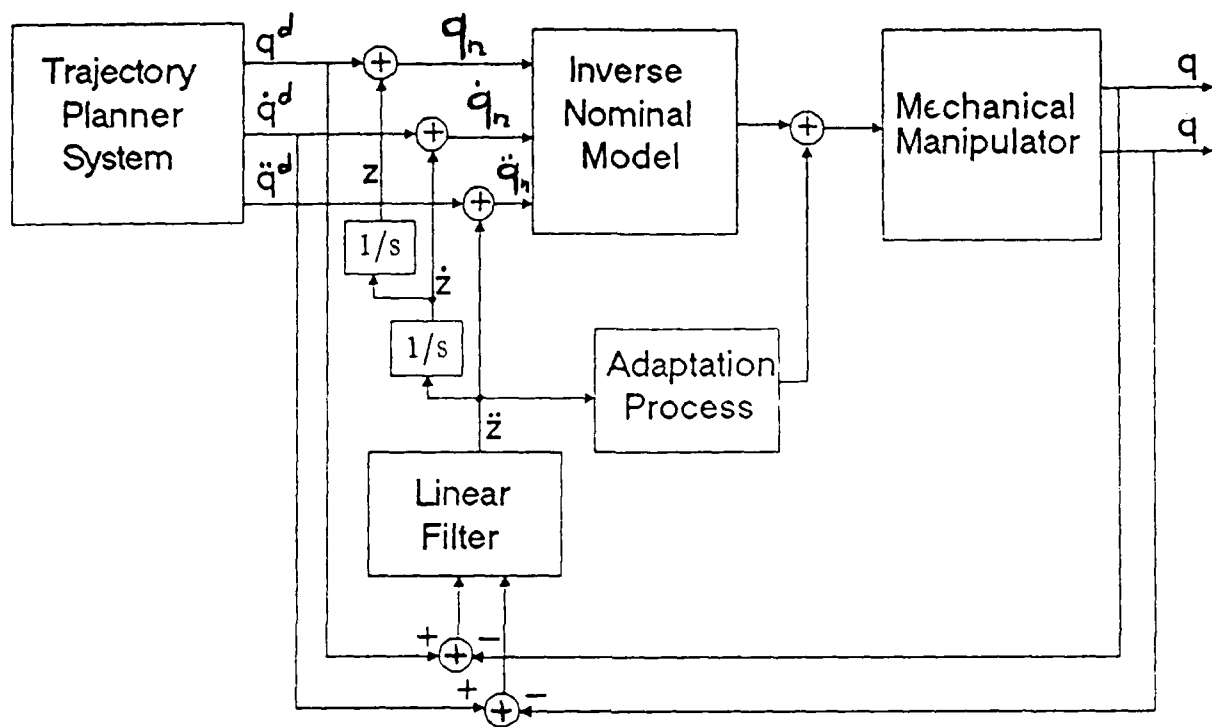


Figure 3.34: Adaptive Nonlinear Model Following Control
(Series Configuration)

In total, the input torques τ are the sum of τ_n and $d\tau$.

$$\tau = \tau_n + d\tau \quad (3.41)$$

where τ_n are the outputs of the recursive algorithm given by:

$$\tau_n(t) = A[q_n(t)]\ddot{q}_n(t) + V[q_n(t), \dot{q}_n(t)] + G[q_n(t)] \quad (3.42)$$

and $d\tau$ are to be determined. From the equations of motion of the manipulator, the total torques τ can also be expressed in terms of the joint angles $q(t)$ as:

$$\tau(t) = A[q(t)]\ddot{q}(t) + V[q(t), \dot{q}(t)] + G[q(t)] \quad (3.43)$$

Subtracting equations (3.43) from (3.42) and substituting $d\tau$ by its expression in equation (3.41) yields:

$$A[q(t)] [\ddot{q}_n(t) - \ddot{q}(t)] = W[q(t), q_n(t), \dot{q}(t), \dot{q}_n(t)] + d\tau \quad (3.44)$$

or, since $A[q(t)]$ is nonsingular,

$$\ddot{q}_n - \ddot{q} = -A^{-1}[W + d\tau] \quad (3.45)$$

where the arguments have been omitted for convenience and where,

$$W = \Delta V[q(t), q_n(t), \dot{q}(t), \dot{q}_n(t)] + \Delta G[q(t), q_n(t)] + \Delta A[q(t), q_n(t)] \ddot{q}_n \quad (3.46)$$

with

$$\Delta V = V[q_n(t), \dot{q}_n(t)] - V[q(t), \dot{q}(t)] \quad (3.47)$$

$$\Delta G = G[q_n(t)] - G[q(t)] \quad (3.48)$$

$$\Delta A = A[q_n(t)] - A[q(t)] \quad (3.49)$$

$$\lim W[q(t), q_n(t), \dot{q}(t), \dot{q}_n(t)] = 0 \quad (3.50)$$

as $q(t) \rightarrow q_n(t)$ and $\dot{q}(t) \rightarrow \dot{q}_n(t)$.

On the other hand,

$$\ddot{q}_n - \ddot{q} = \ddot{q}^d + L^{-1}(s^2 F(s) E(s)) - \ddot{q} \quad (3.51)$$

That is,

$$\bar{q}_n - \bar{q} = \ddot{e} + L^{-1}(s^2 F(s) E(s)) \quad (3.52)$$

where $L^{-1}(\cdot)$ is the inverse Laplace transform of the given function. Substituting the expression of $(\bar{q}_n - \bar{q})$ from equation (3.52) into equation (3.45) we obtain:

$$\ddot{e} + L^{-1}(s^2 F(s) E(s)) = -A^{-1}[W + d\tau] \quad (3.53)$$

Equation (3.53) shows that the adjusting forward signals are crucial to the stability of the system. When these signals are not used, such as in [51], the error equation is unstable.

Any stable filter $F(s)$ of the form:

$$F(s) = \frac{a_{n-2}s^{n-2} + a_{n-3}s^{n-3} + \dots + a_0}{b_n s^n + b_{n-1}s^{n-1} + \dots + b_0} \quad (3.54)$$

will yield the desired error equation in (3.39). A more interesting choice, however, is:

$$F(s) = \frac{1}{s^2} (K_v s + K_p) \quad (3.55)$$

where K_v and K_p are velocity and position feedback matrices, respectively. The motive behind choosing $F(s)$ as in equation (3.55) lies in the fact that it relaxes the computation considerably. This can be seen by evaluating $z(t)$, $\dot{z}(t)$, and $\ddot{z}(t)$ corresponding to this choice:

$$\begin{aligned} \ddot{z}(t) &= L^{-1}(s^2 F(s) E(s)) \\ \ddot{z}(t) &= K_v \dot{e}(t) + K_p e(t) \end{aligned} \quad (3.56)$$

which is actually no more than velocity and position measurements feedback. The quantities $\dot{z}(t)$ and $z(t)$ can then be obtained from the above expression of $\ddot{z}(t)$ by a simple and a double integration, respectively. With $F(s)$ selected as in equation (3.55), the error equation in (3.53) becomes:

$$\ddot{e} + K_v \dot{e} + K_p e = -A^{-1}[W + d\tau] \quad (3.57a)$$

Equation (3.57a) is an equivalent error model representation of the proposed adaptive control law. It can be partitioned into a linear time invariant system connected with a

nonlinear time varying block in the feedback as shown in Figure 3.35. It should be clear that K_v and K_p are chosen such that the forward transmission function is stable.

Notice that, if a filter of the type given in equation (3.54) is used instead, the term is decomposed as:

$$F(s) = 1 - F'(s)$$

Hence, equation (3.53) becomes:

$$\ddot{e} + K_v \dot{e} + K_p e = -A^{-1} [W_1 + d\tau] \quad (3.57b)$$

with

$$W_1 = W + L^{-1}(F'(s)E(s))$$

which is a more general form of equation (3.57a).

It remains now to determine the control inputs $d\tau$ based on the knowledge of some upper bounds of:

$$\left\| W [q(t), q_n(t), \dot{q}(t), \dot{q}_n(t)] \right\|$$

such that $\|e(t)\| < \epsilon$, for $t \rightarrow \infty$ and for any initial conditions. Here ϵ is a small number.

Let

$$\begin{aligned} \eta(t) &= \bar{z}(t) = L^{-1}(s^2 F(s) E(s)) \\ \eta(t) &= K_v \dot{e}(t) + K_p e(t) \end{aligned} \quad (3.58)$$

The quantity $\dot{\eta}(t)$ is given by:

$$\dot{\eta}(t) = -K_v [a\eta + A^{-1} [W + d\tau]] \quad (3.59)$$

with $a = K_v^{-1} K_p = \text{diag}(a_i)$, $a_i > 0$ for $i = 1, \dots, n$, since

$K_v = \text{diag}(k_{vi})$, $k_{vi} > 0$ and $K_p = \text{diag}(k_{pi})$, $k_{pi} > 0$.

Define $d\tau$ as:

$$d\tau(t) = \ell(t) \frac{\eta(t)}{\|\eta(t)\|} \quad (3.60)$$

with

$$\ell(t) \geq \lambda_{\max}(A) \|W^T A^{-1}\| \quad (3.61)$$

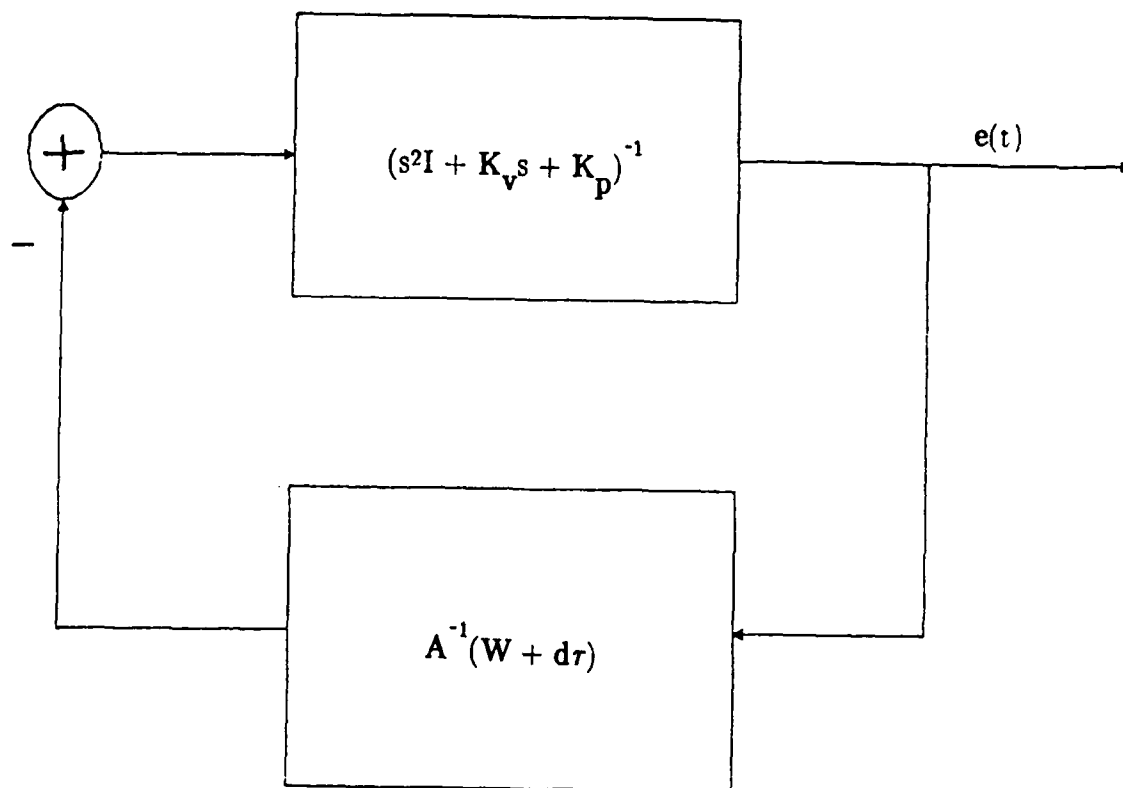


Figure 3.35: Error model representation of the proposed adaptive control law

FACT:

Equations (3.60) and (3.61) imply:

$$\lim_{t \rightarrow \infty} \eta(t) = 0 \quad (3.62)$$

Consequently,

$$\lim_{t \rightarrow \infty} e(t) = 0 \quad (3.63)$$

$$\lim_{t \rightarrow \infty} \dot{e}(t) = 0 \quad (3.64)$$

$$\lim_{t \rightarrow \infty} (q_n - q^d) = 0 \quad (3.65)$$

PROOF:

The above claim can be proven by choosing a Lyapunov function $V(\eta)$ as follows:

$$V(\eta) = \frac{1}{2} \eta^T(t) \eta(t) \quad (3.66)$$

Then

$$\dot{V}(\eta) = \dot{\eta}^T(t) \eta(t) \quad (3.67)$$

$$\dot{V}(\eta) = - \left[\eta^T \alpha K_v \eta + W^T A^{-1} K_v \eta + d \tau^T A^{-1} K_v \eta \right] \quad (3.68)$$

Notice that since $\alpha = \text{diag}(\alpha_i)$, $\alpha_i > 0$ and $K_v = \text{diag}(k_{vi})$, $k_{vi} > 0$, then $\alpha K_v = \beta = \text{diag}(\beta_i)$, $\beta_i > 0$.

Now replace $d\tau(t)$ by their expressions in (3.60) to obtain:

$$\dot{V}(\eta) = - \eta^T \beta \eta - \left[W^T A^{-1} K_v \eta + \ell \eta^T A^{-1} K_v \frac{\eta}{\|\eta\|} \right] \quad (3.69)$$

If we select $\ell(t)$ so that:

$$\lambda_{\min}(A^{-1}) \ell > \|W^T A^{-1}\| \quad (3.70)$$

then

$$\dot{V}(\eta) \leq - \eta^T \beta \eta \leq - \beta_{\min} \eta^T(t) \eta(t) \quad (3.71)$$

That is,

$$\dot{V}(\eta) \leq - 2\beta_{\min} V(\eta) \quad (3.72)$$

with

$$\beta_{min} = \min(\beta_i), \text{ for } i = 1, \dots, n.$$

by which (3.62), (3.63), (3.64) and (3.65) are satisfied.

The proof for the case where the filter in (3.54) is used, follows along the same lines.

QED

The above control law is simulated on the two link arm model. The following values of ℓ , K_p , and K_v are used:

$$\ell = \begin{bmatrix} 16 & 0 \\ 0 & 10 \end{bmatrix}$$

$$K_p = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

$$K_v = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

When 10% error in the load is assumed, the tracking quality, when no feedback corrections are applied, is poor in both the first (Figure 3.36) and the second (Figure 3.37) links. The joint errors are of the order of 6° and 17° , respectively as shown in Figure 3.38. The torque signals (Figure 3.39) are similar to the ones obtained from the inverse dynamics law. There is no noticeable improvement (or little) in the tracking quality of both the first (Figure 3.40) and the second (Figure 3.41) links, when the adjusting signals $z(t)$, $\dot{z}(t)$, and $\ddot{z}(t)$ alone are used. This can be seen from the plots of the time evolution of the tracking errors in Figure 3.42, and of the input torques in Figure 3.43. However, the quality of the tracking is improved drastically in both the first (Figure 3.44) and the second (Figure 3.45) links, when both the adjusting signals and the correction torques are implemented. Figure 3.46 shows that the tracking errors in both links are practically reduced to zero. This is about 8° better than [51] and 10° better than [63]. As expected, figure 3.47 shows a moderate chattering, within a very acceptable range of values, in the control signals. Because integrators are used to generate $z(t)$ and $\dot{z}(t)$, it is of interest to check that the presence of

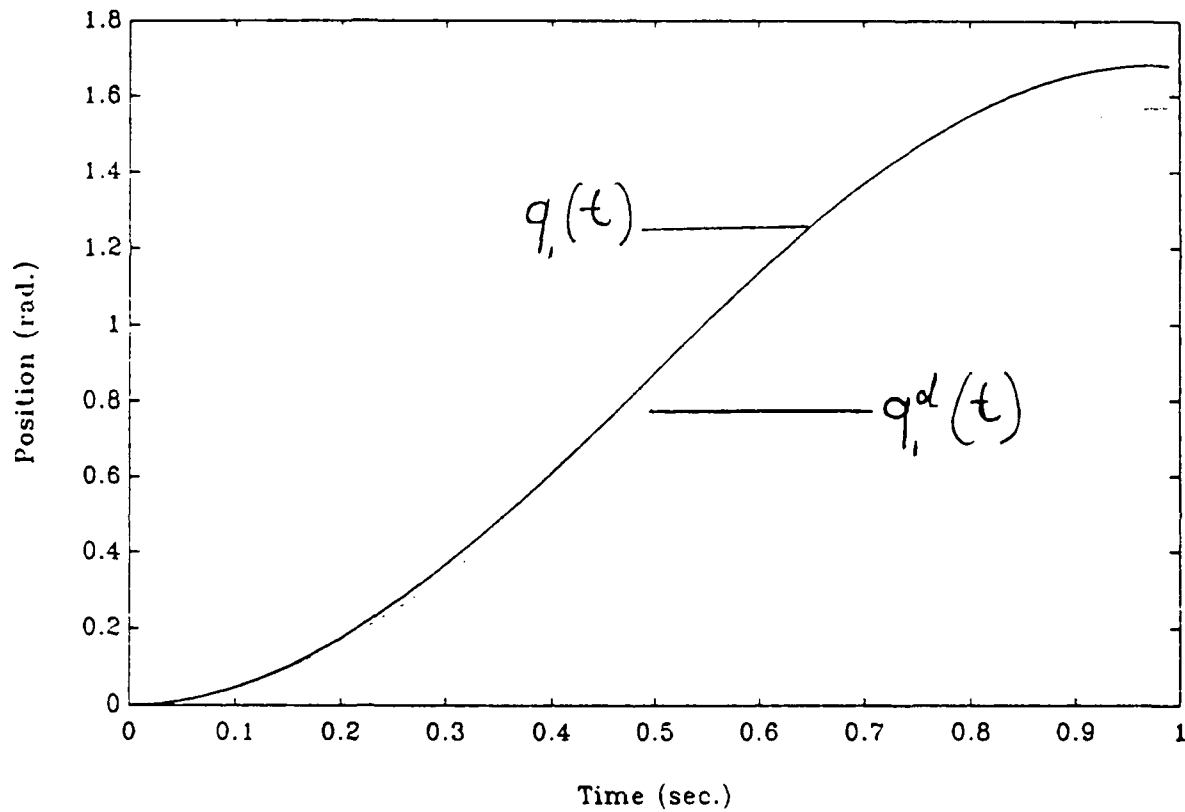


Figure 3.36: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (No Feedback)

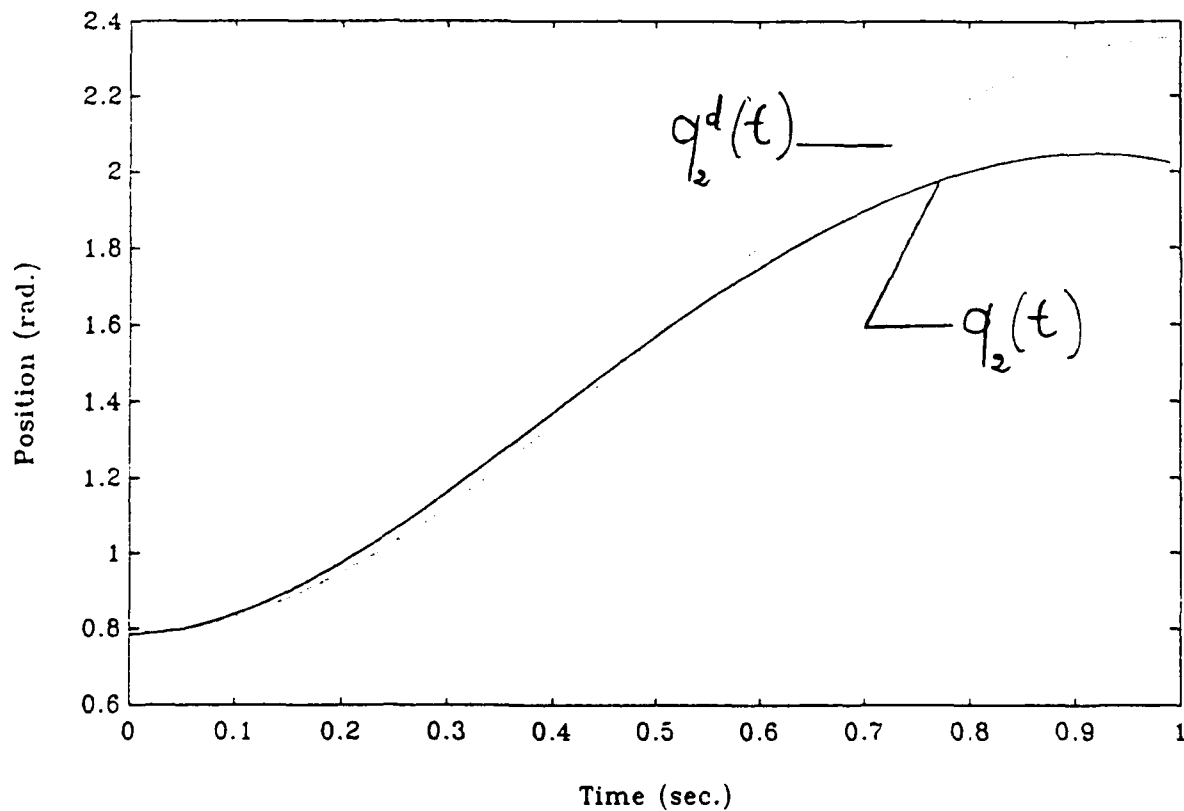


Figure 3.37: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (No Feedback)

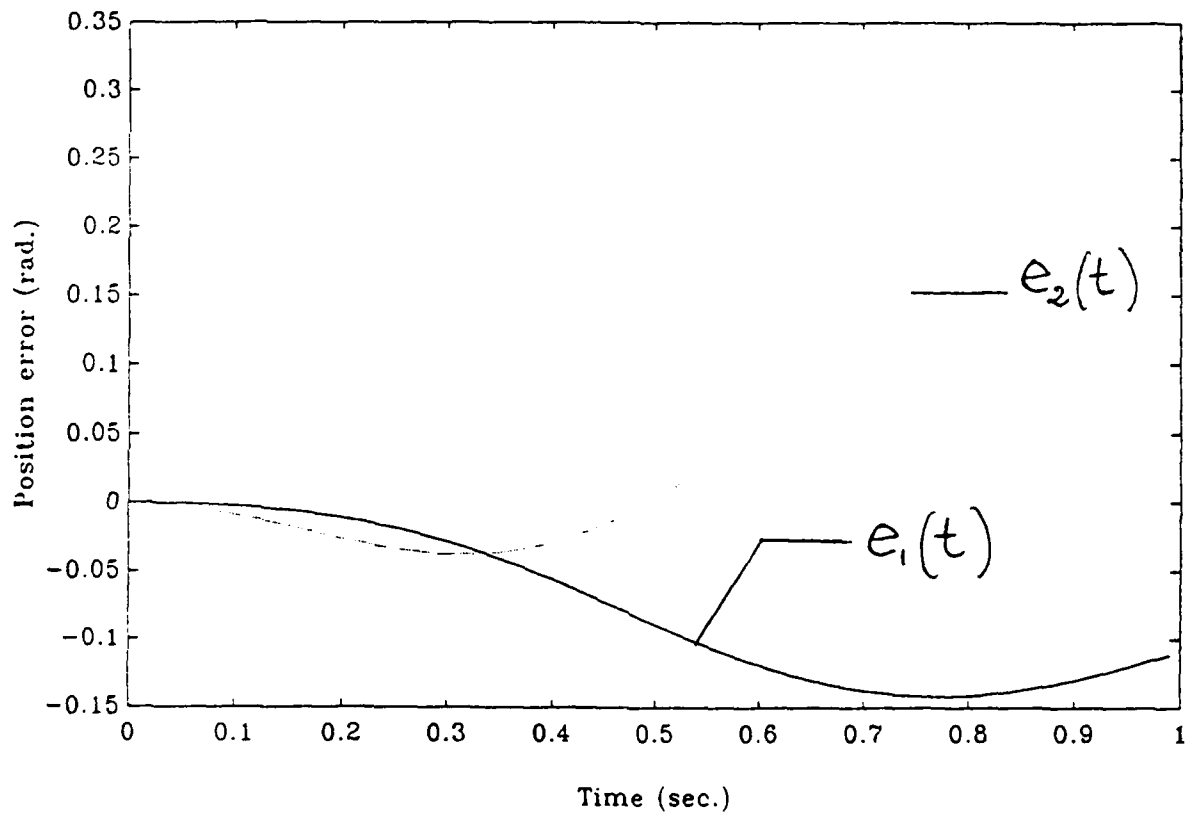


Figure 3.38: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (No Feedback)

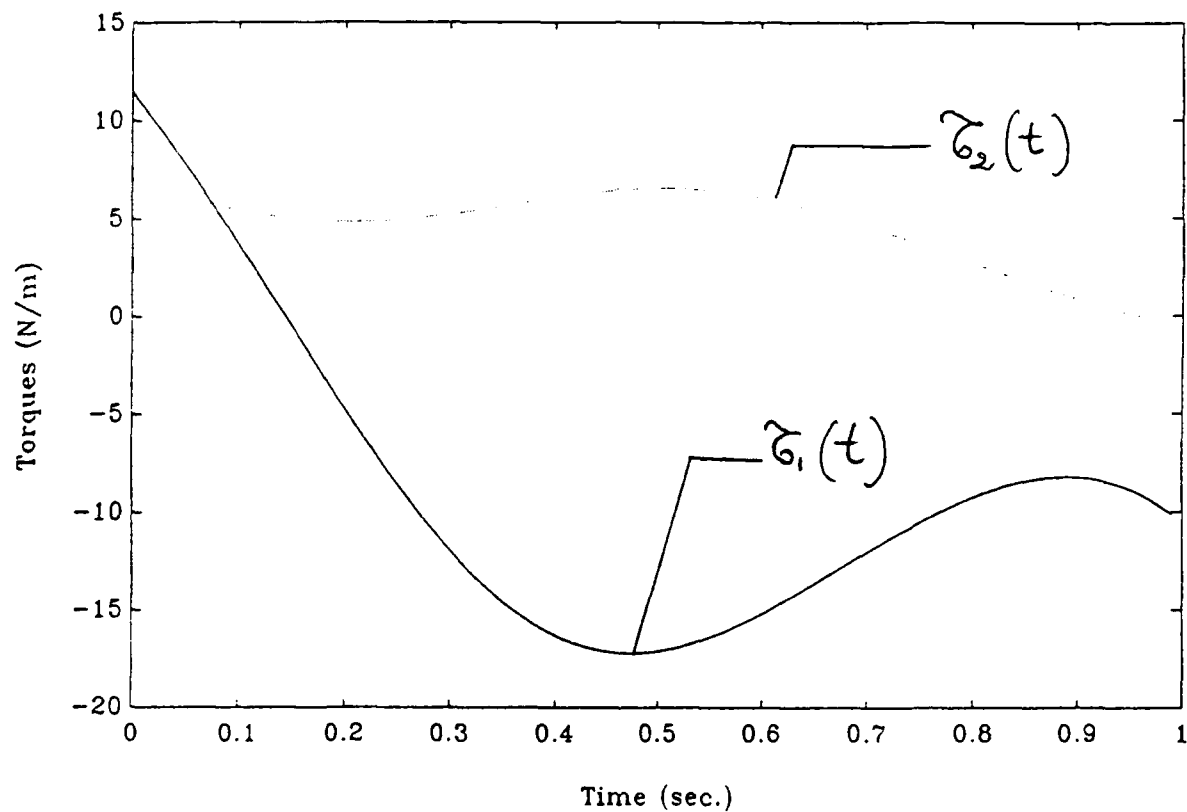


Figure 3.39: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric Uncertainties only (No Feedback)

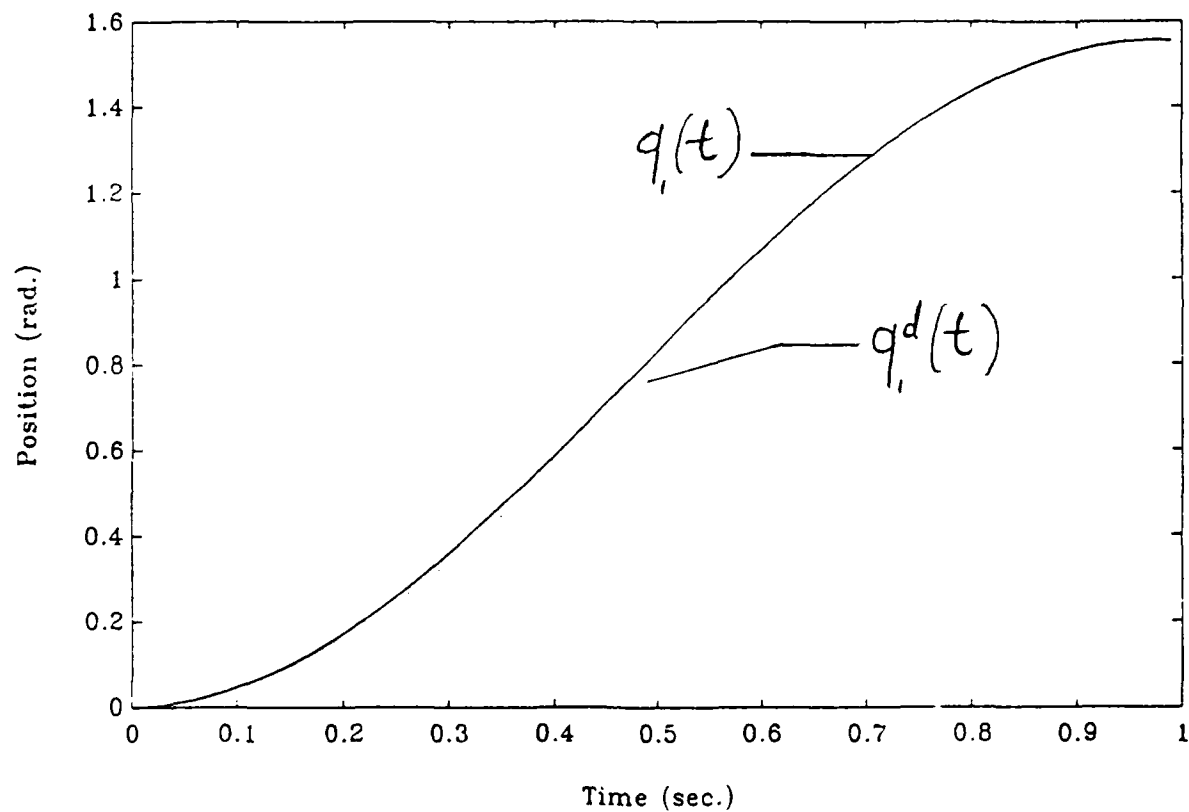


Figure 3.40: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (Using Adjusting Signals alone)

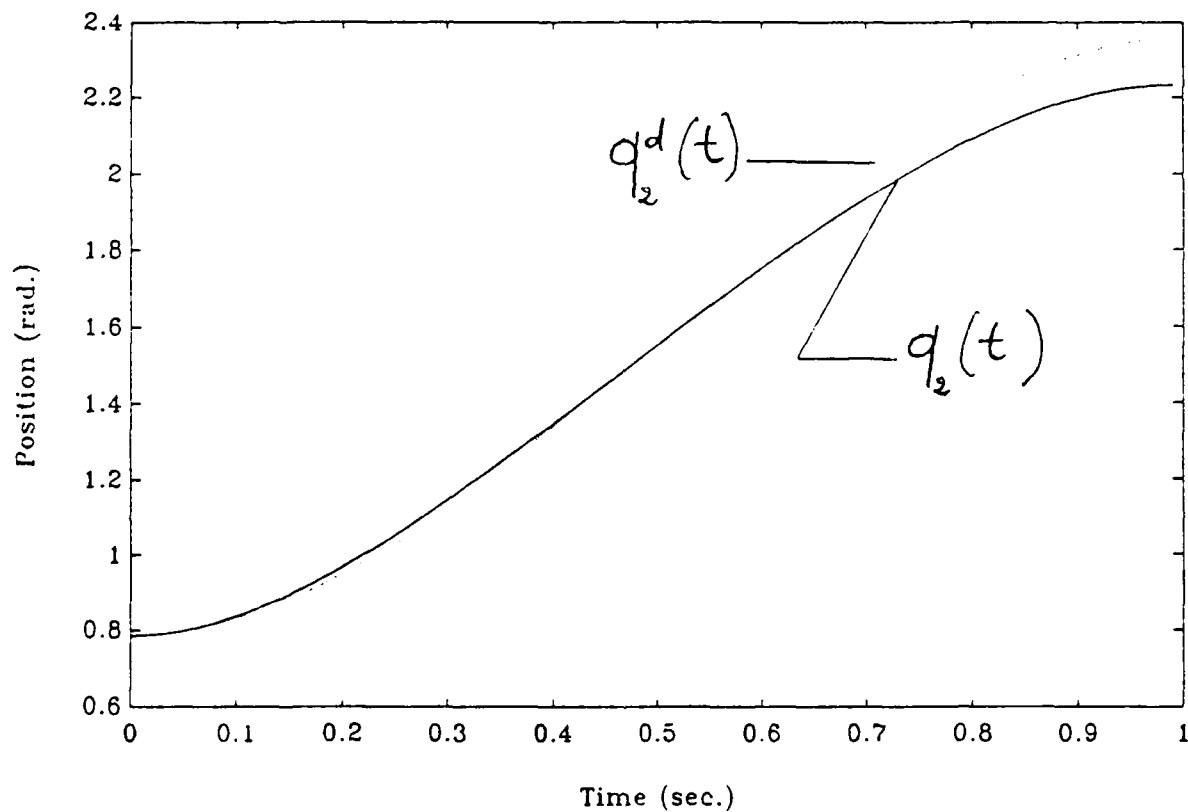


Figure 3.41: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (Using Adjusting Signals alone)

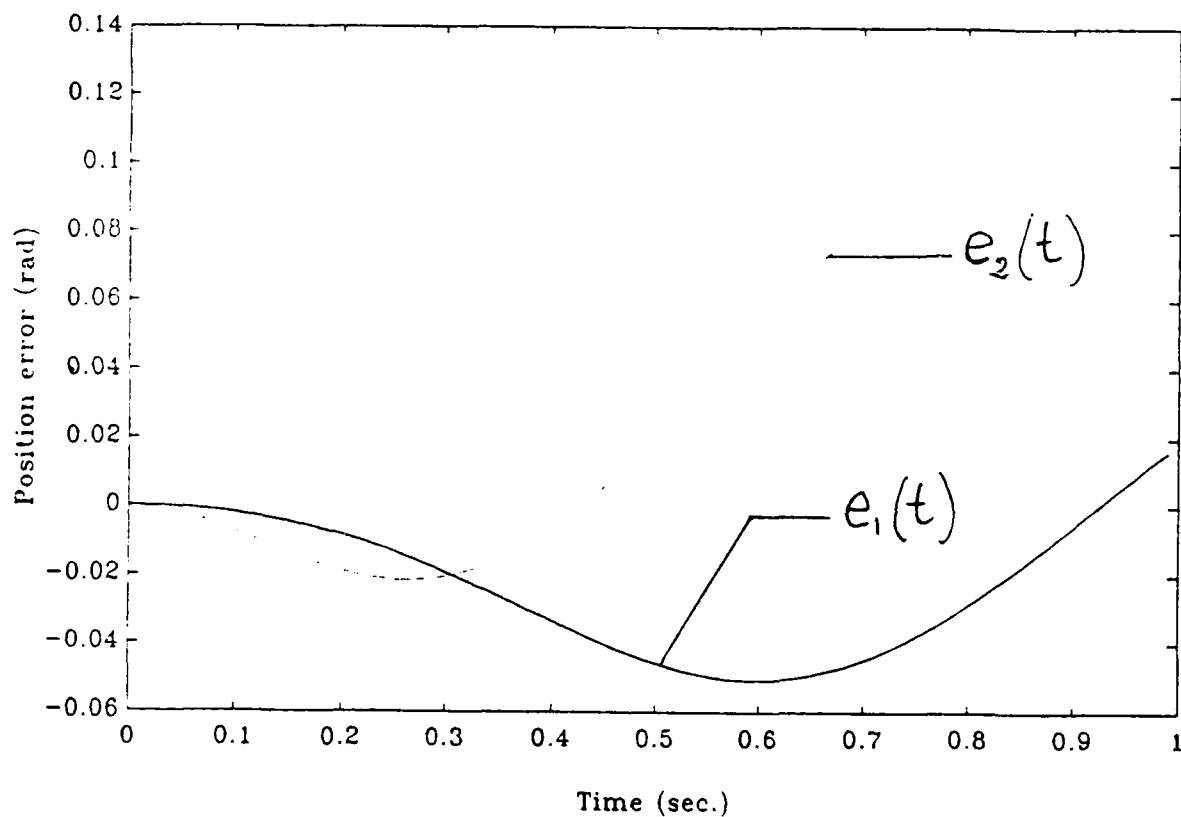


Figure 3.42: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (Using Adjusting Signals alone)

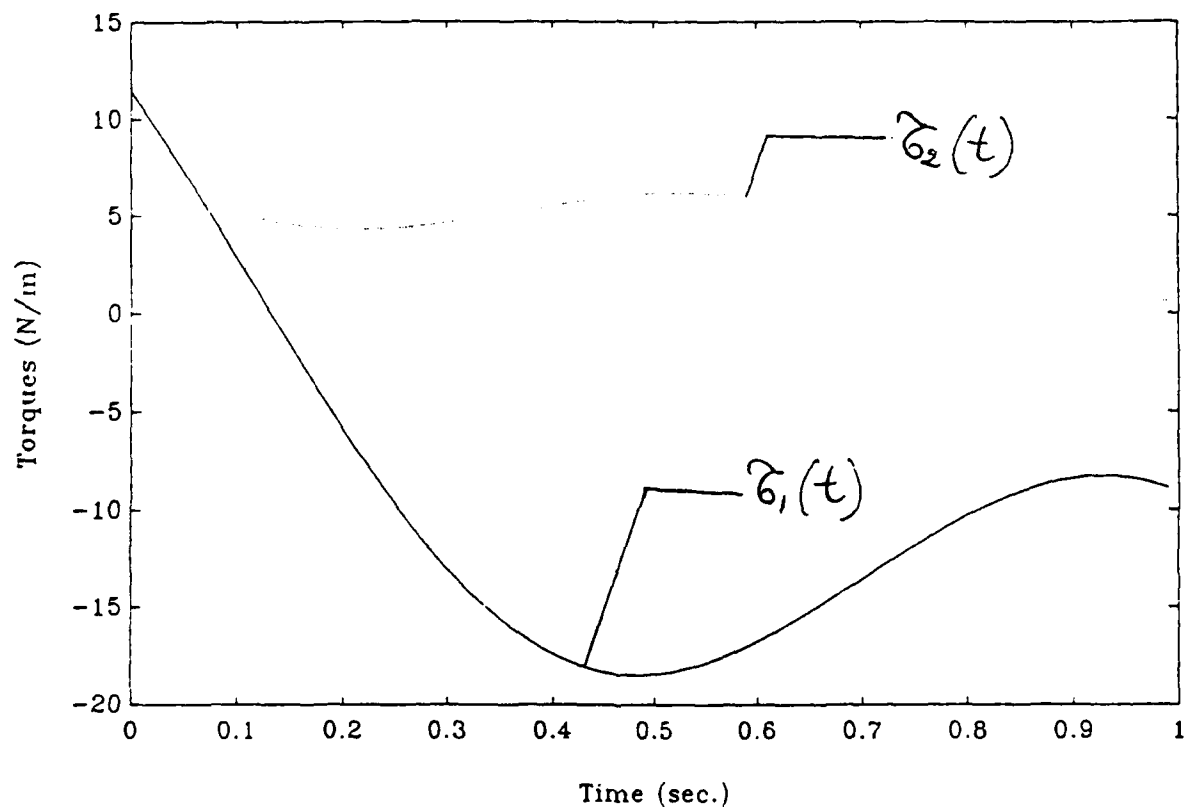


Figure 3.43: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric Uncertainties only (Using Adjusting Signals alone)

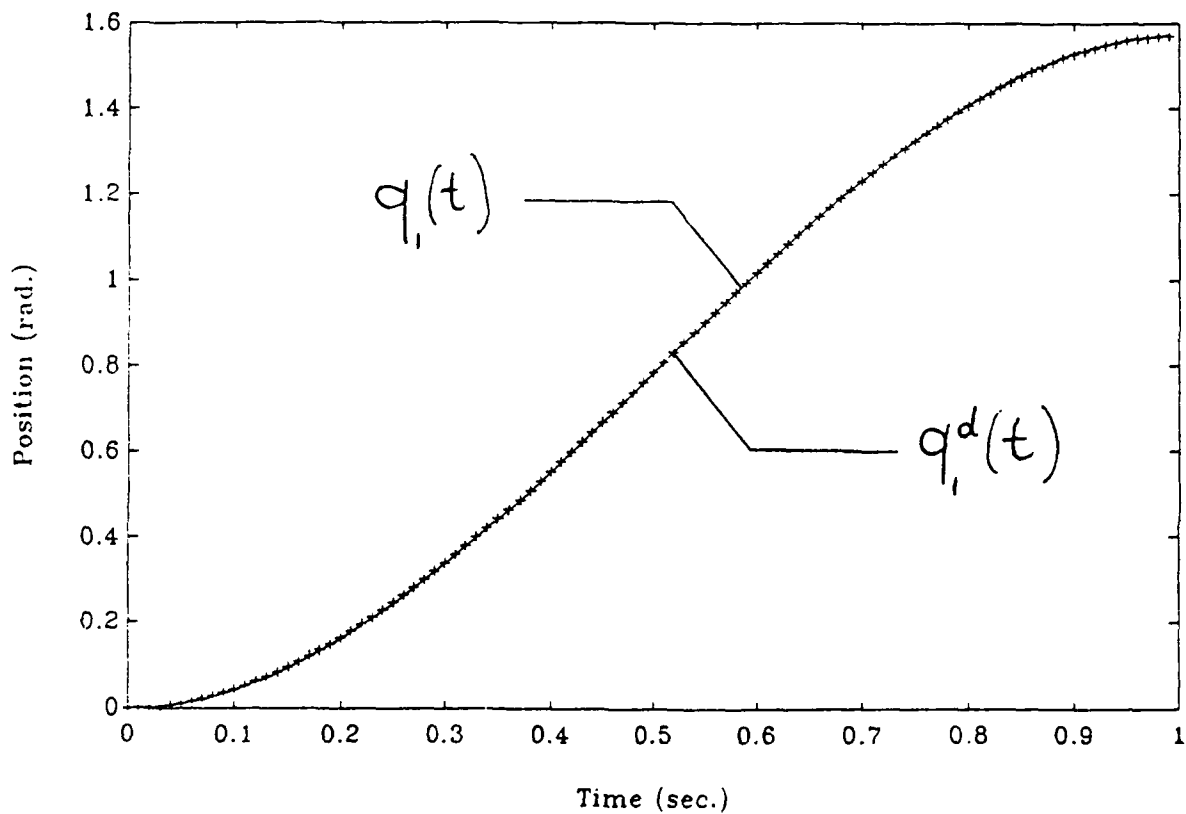


Figure 3.44: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With Adjusting Signals and Correcting Torques)

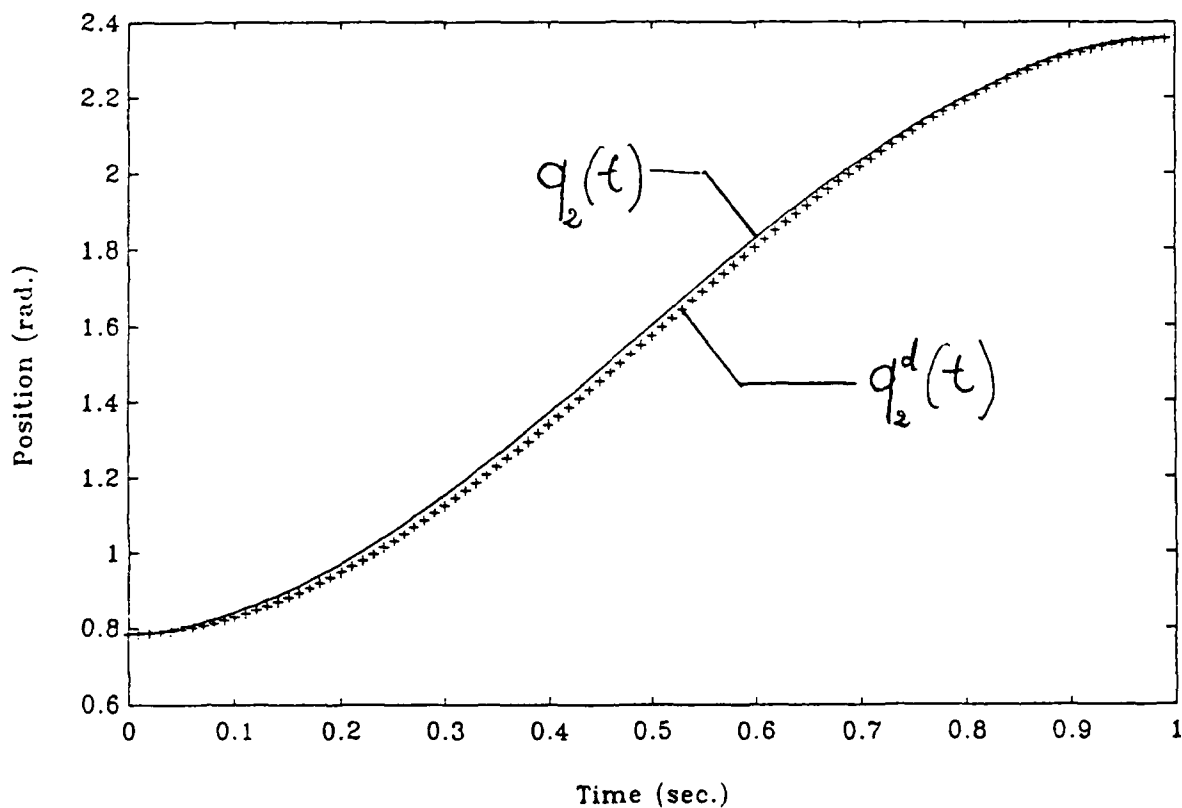


Figure 3.45: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With Adjusting Signals and Correcting Torques)

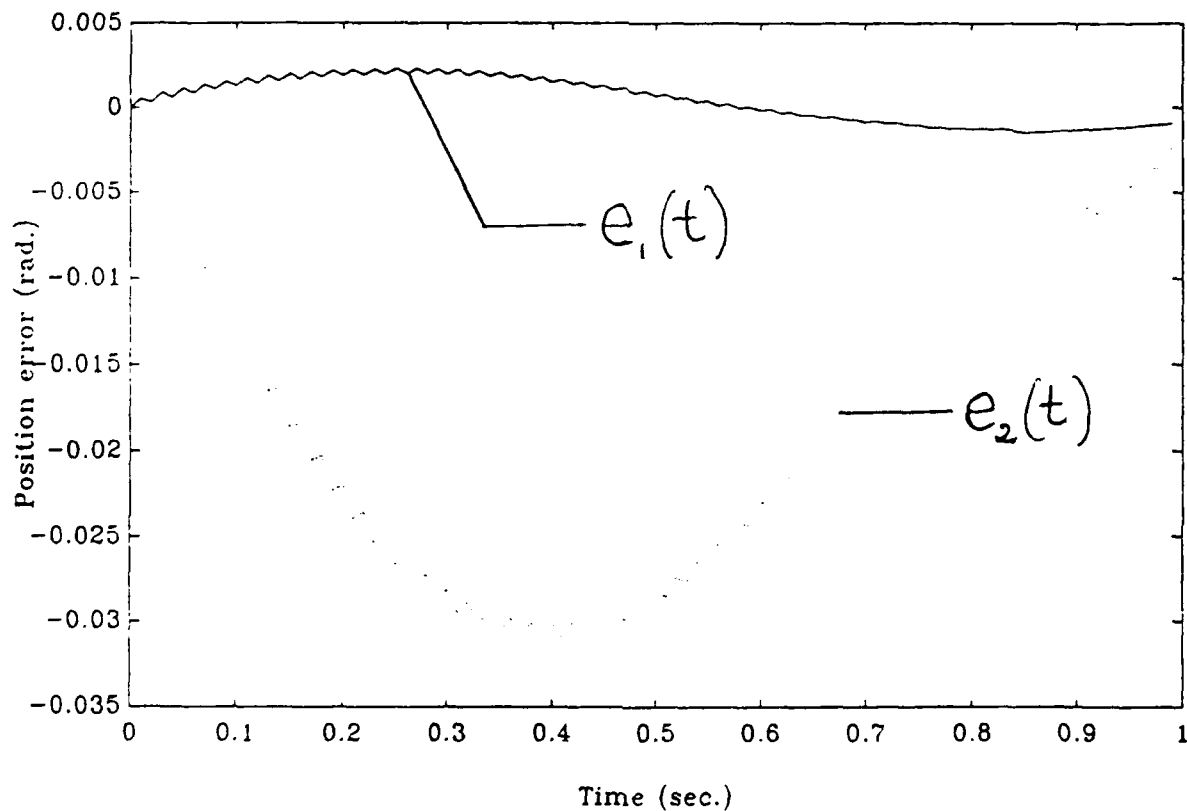


Figure 3.46: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With Adjusting Signals and Correcting Torques)

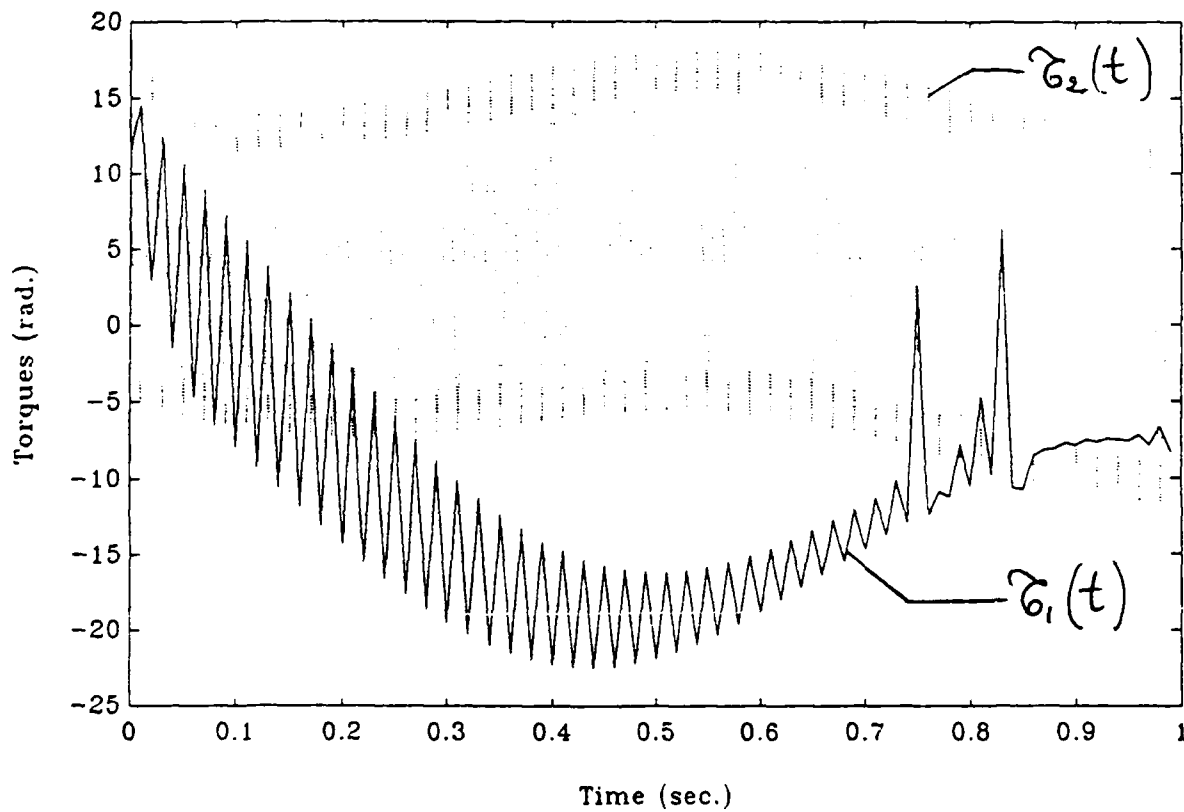


Figure 3.47: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric Uncertainties only (With Adjusting Signals and Correcting Torques)

offset measurements in such signals does not affect the tracking quality of the overall system. Simulations results with initial conditions of $e_1 = e_2 = 115^\circ$ are given in Figures 3.48 through 3.51. There is practically no degradation in the tracking quality of either the first (Figure 3.48) or the second link (Figure 3.49). The tracking errors (Figure 3.50) and the input torques (Figure 3.51) are as before. To show the robustness of this methodology to parameter disturbances, 50% error in the load is assumed and a term of the form $F = C\text{sgn}(\dot{q}_i) + V\dot{q}_i$ is added to the plant as unmodelled friction, where C and V are Coulomb friction and viscous friction constants, respectively. The simulation results from this case with $C = V = 4$ are reported in Figures 3.52 through 3.63. The actual trajectories (q_1 and q_2) diverge considerably from the desired trajectories (q_1^d and q_2^d) when no feedback is used as seen in Figures 3.52 and 3.53, respectively. The tracking errors are in the order of 32° in the first link and of 114° in the second link as shown in Figure 3.54. The required input torques are reported in Figure 3.55. A considerable improvement in the tracking quality of both links is achieved even when only the adjusting signals alone are used as can be seen in Figures 3.56 and 3.57, respectively. However, the joint errors are still of the order of 12° in the first link and of 23° in the second link as seen from Figure 3.58. The control signals are reported in Figure 3.59. When both the adjusting signals and the feedback correcting torques are implemented, the tracking quality is close to perfect as can be seen in Figure 3.60, for the first link and in Figure 3.61, for the second link. The errors in both links (Figure 3.62) are drastically reduced to approximately 0.3° and 0.1° , respectively; while the input torques (Figure 3.63) remain at very acceptable range of magnitudes with mild chattering. When these same conditions are simulated with the Adaptive Perturbation Control Law of [51], the quality of the tracking is poor for both the first (Figure 3.64) and the second (Figure 3.65) links. Figure 3.66 shows the time variations of such errors. These errors are of an order of magnitude of 15° in the first link and of 12° in the second link. Respectively, this is about 50 and 120 times **less accurate** than the proposed approach (see

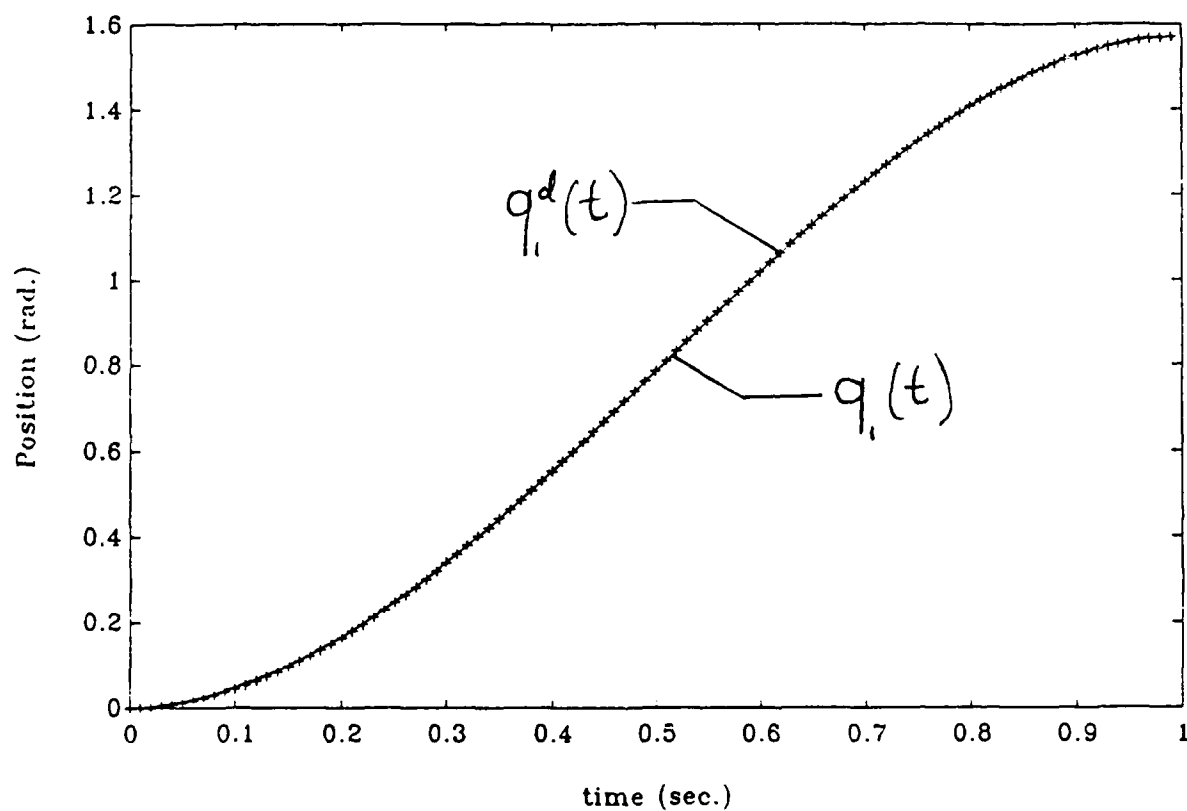


Figure 3.48: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With 115° offset)

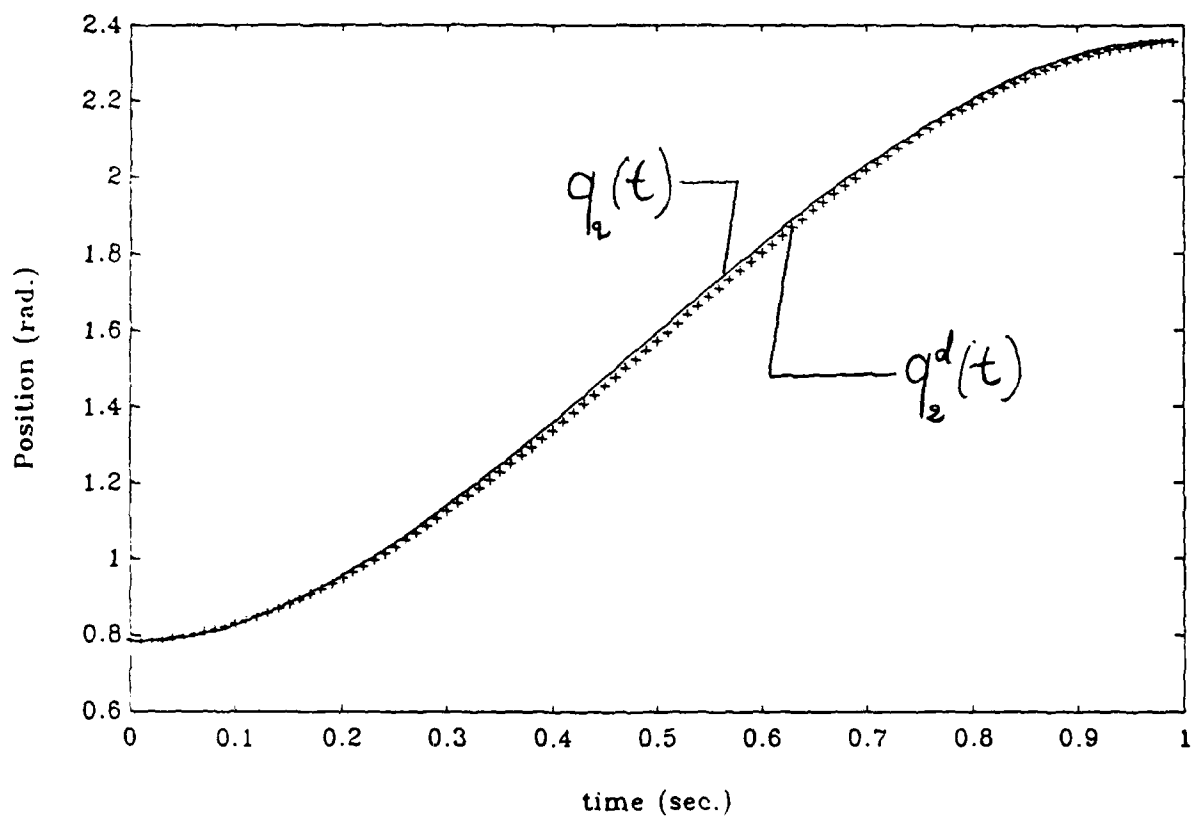


Figure 3.49: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With 115° offset)

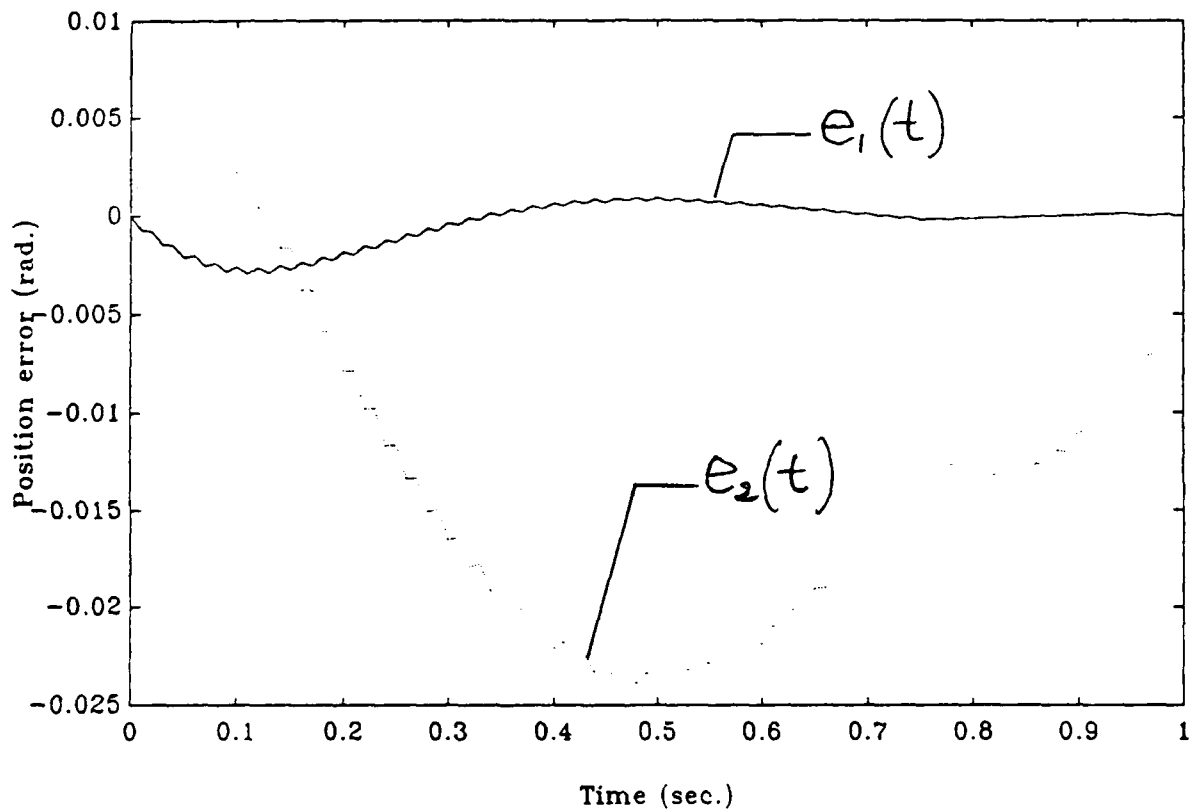


Figure 3.50: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric Uncertainties only (With 115° offset)

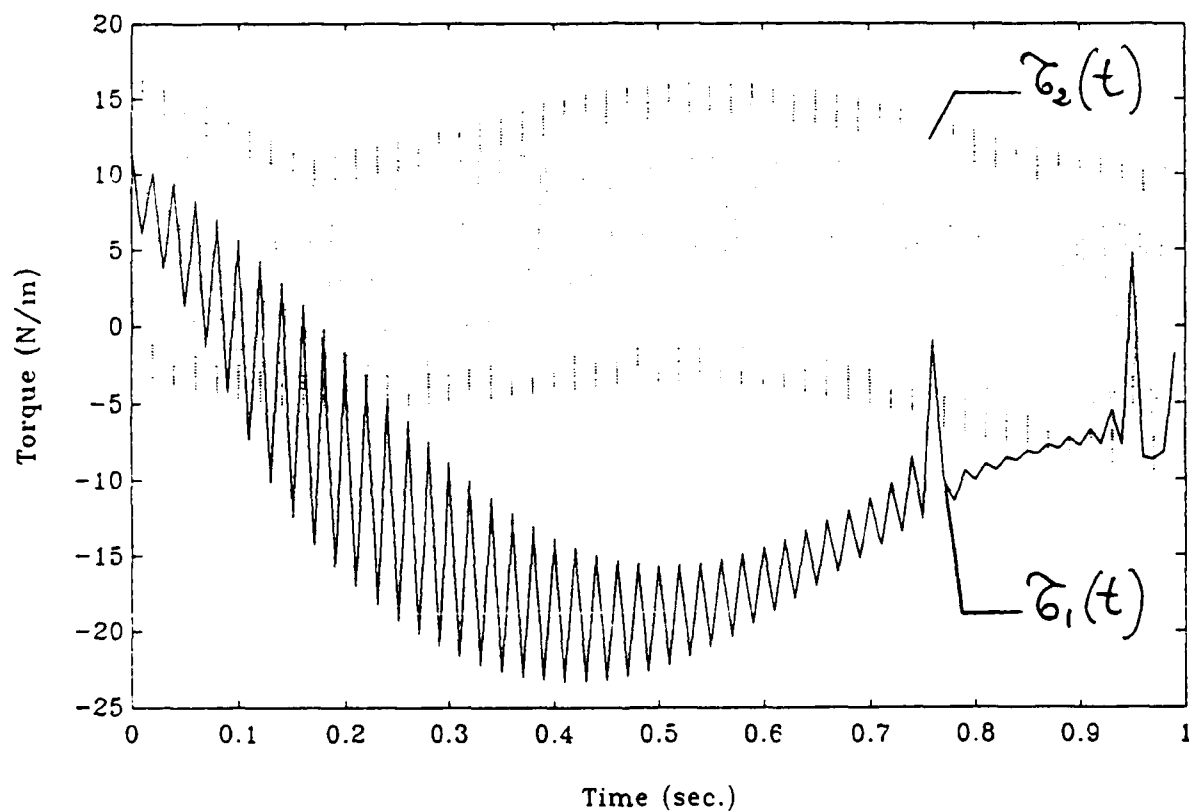


Figure 3.51: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric Uncertainties only (With 115° offset)

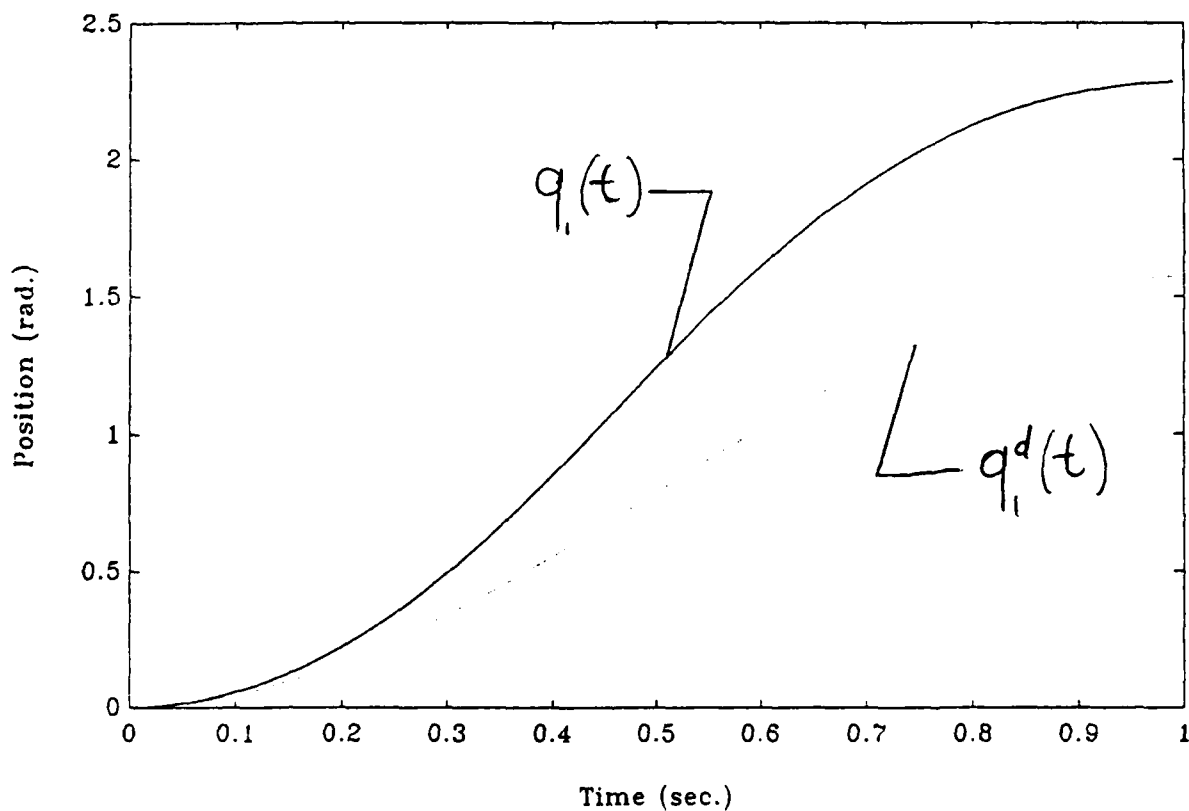


Figure 3.52: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (No Feedback)

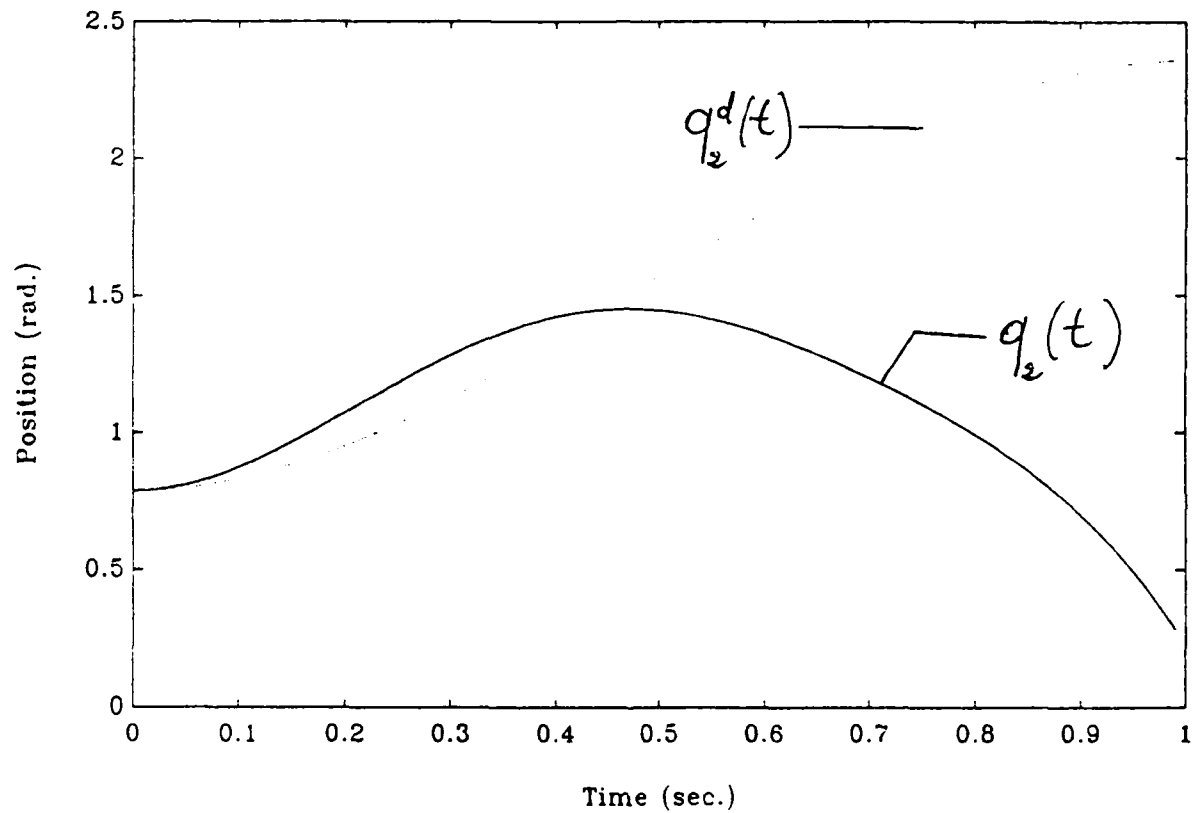


Figure 3.53: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (No Feedback)

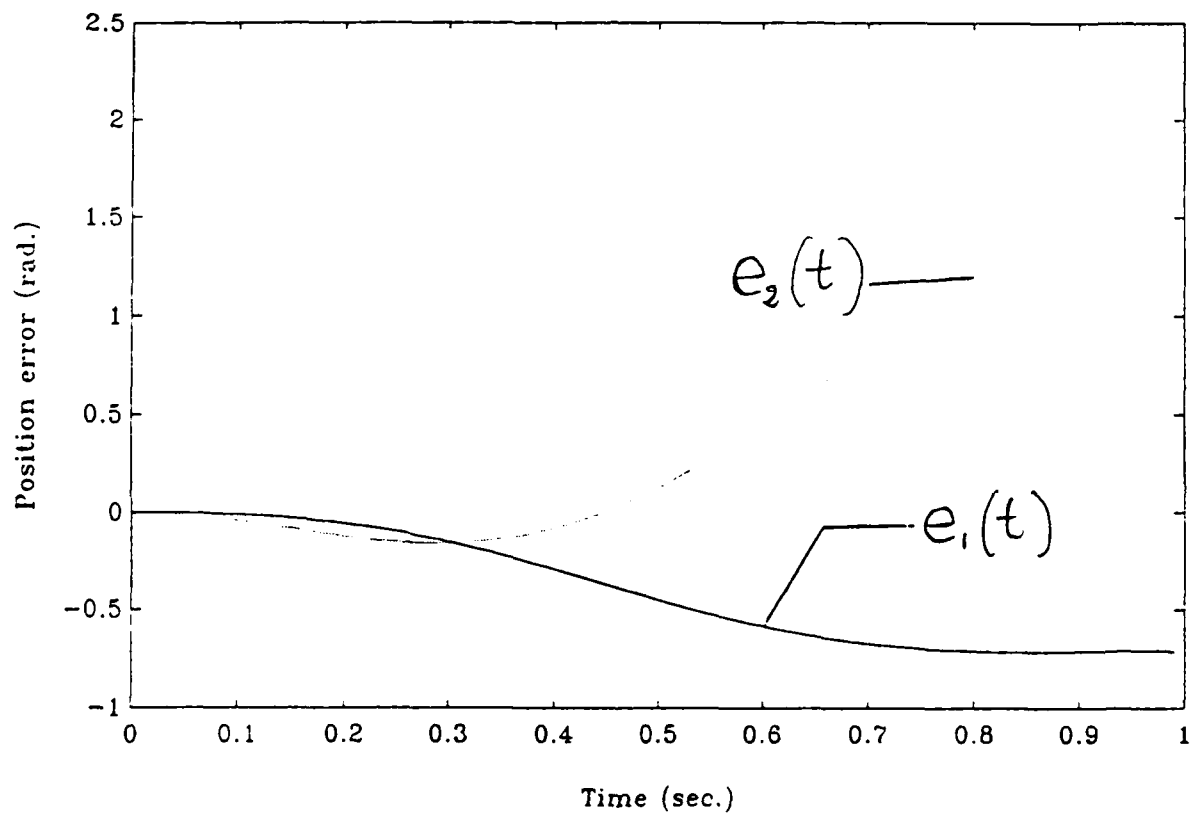


Figure 3.54: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (No Feedback)

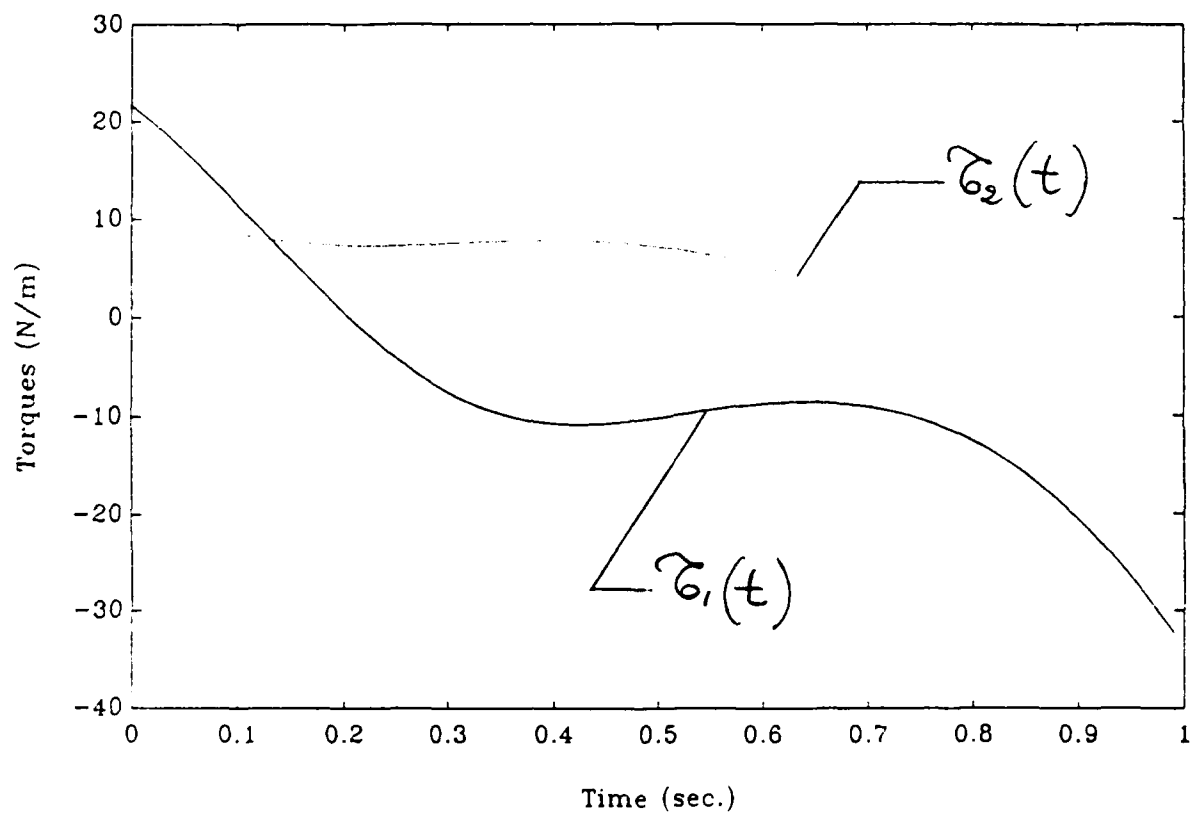


Figure 3.55: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (No Feedback)

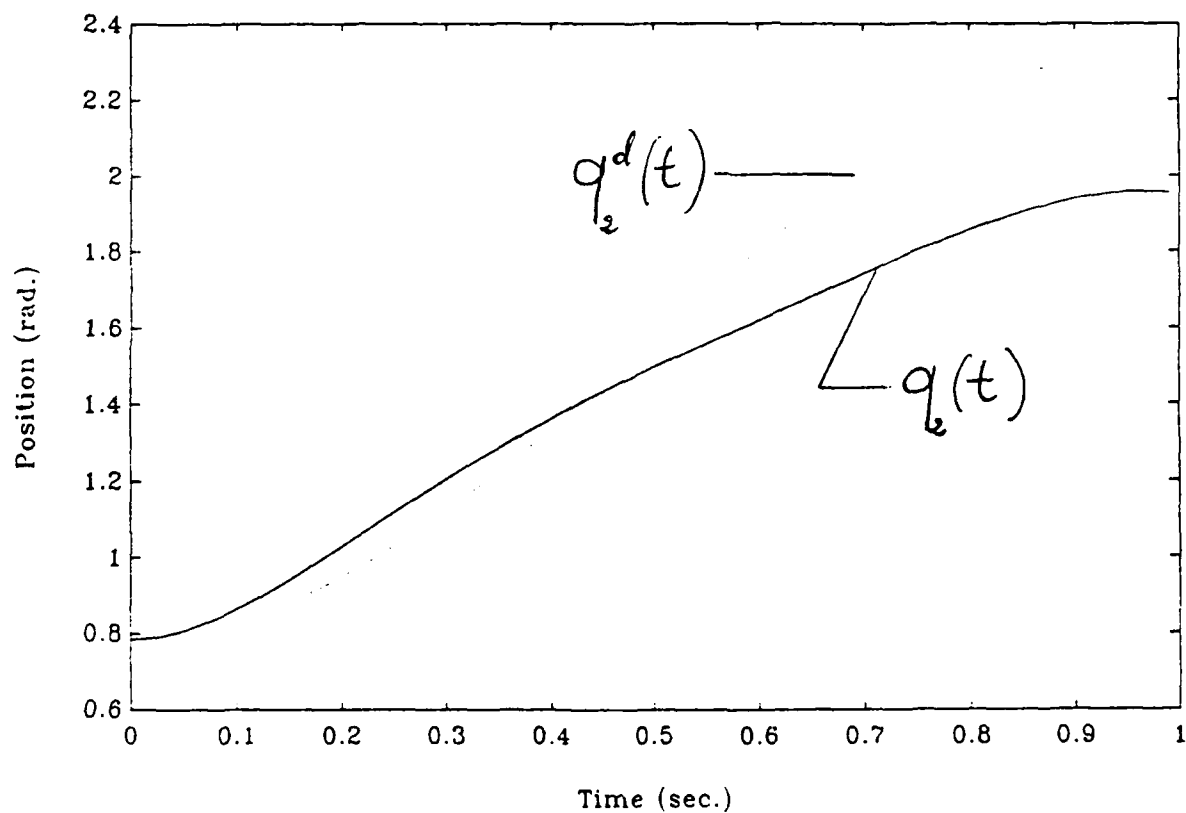


Figure 3.57: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (With $d\hat{\zeta} = 0$)

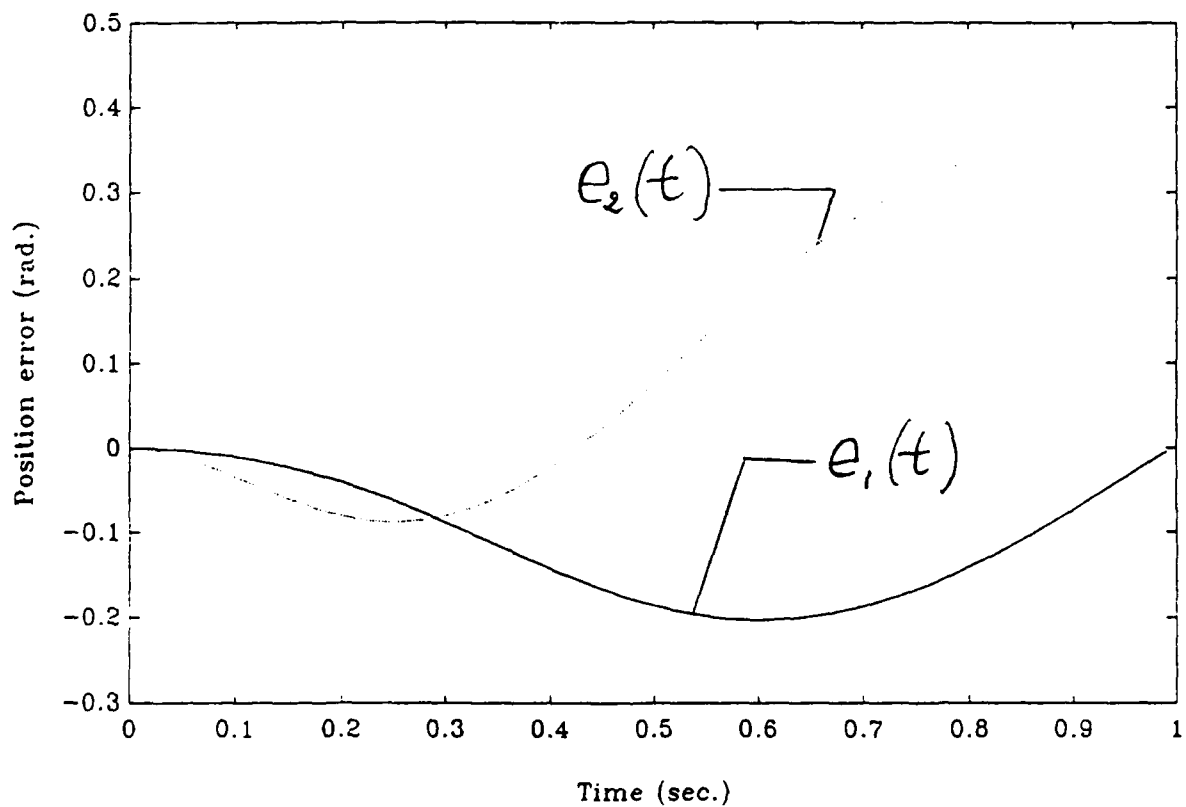


Figure 3.58: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (With $d\tilde{\zeta} = 0$)

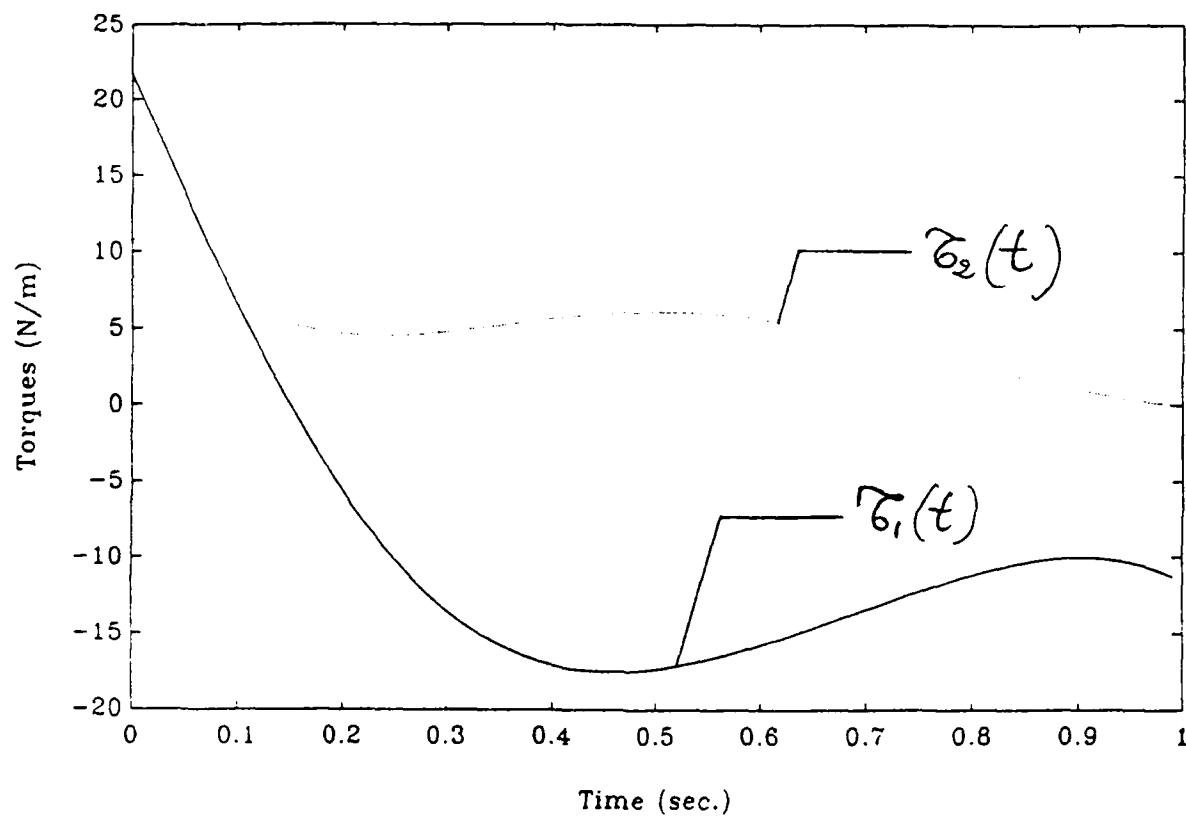


Figure 3.59: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (With $d\hat{\theta} = 0$)

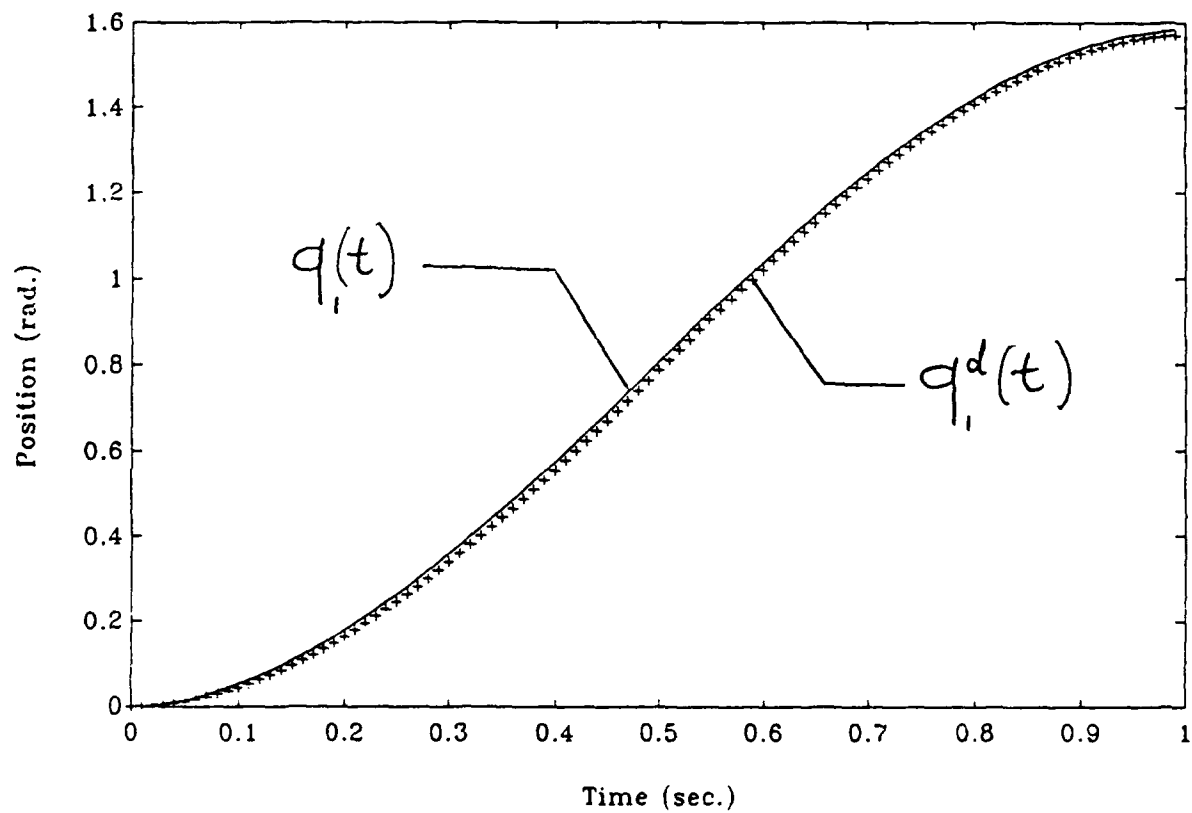


Figure 3.60: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Complete Scheme)

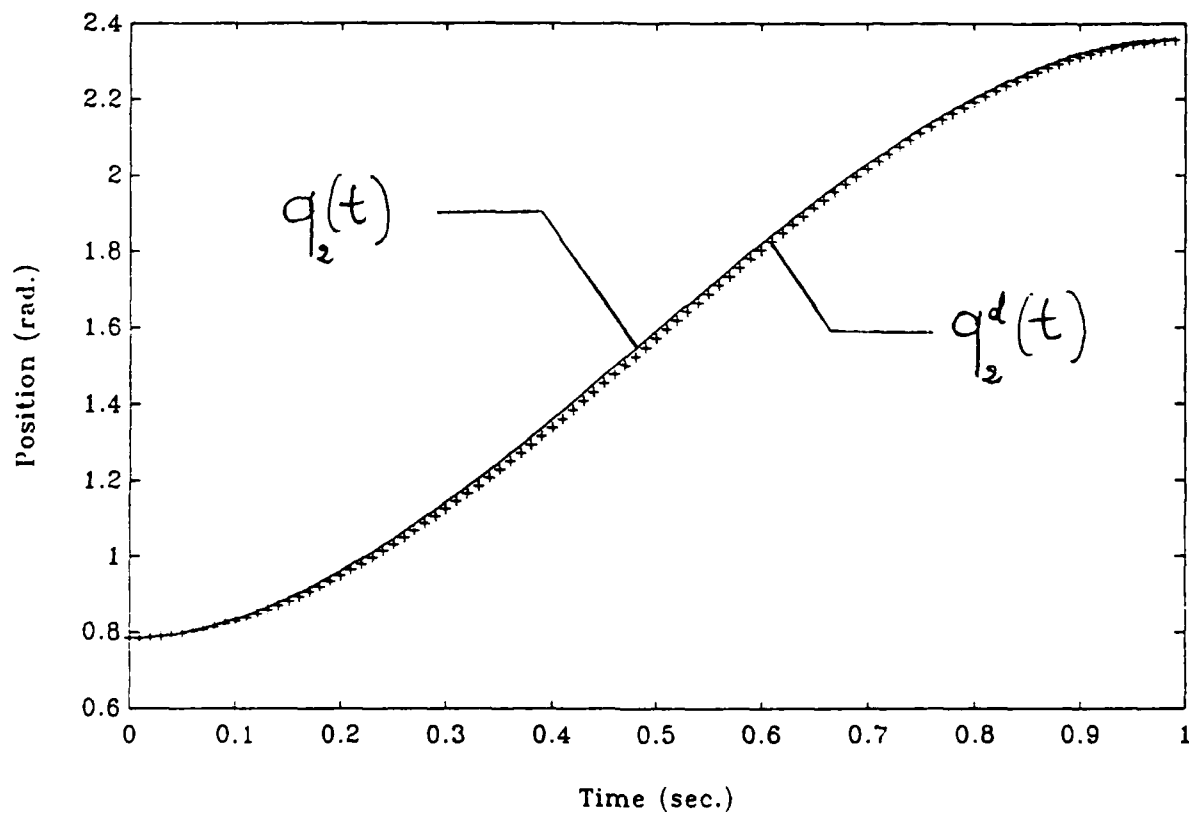


Figure 3.61: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Complete Scheme)

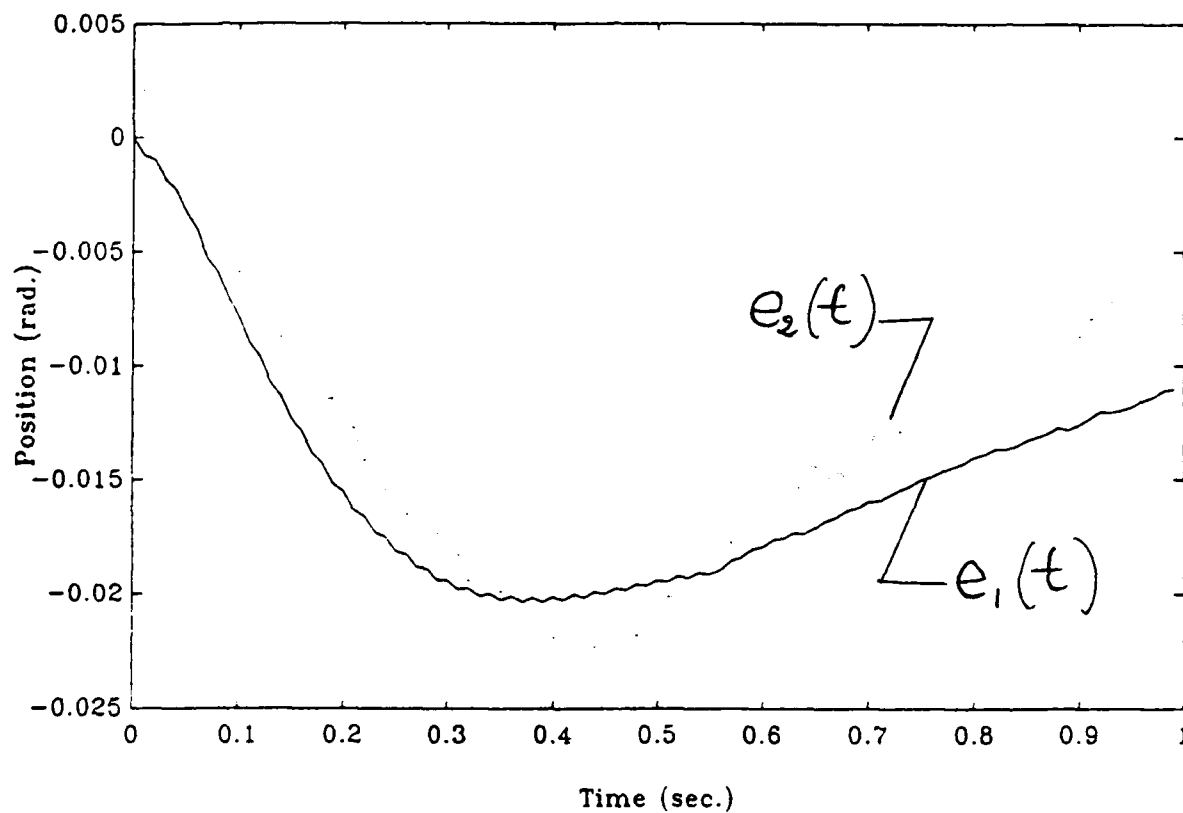


Figure 3.62: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Complete Scheme)

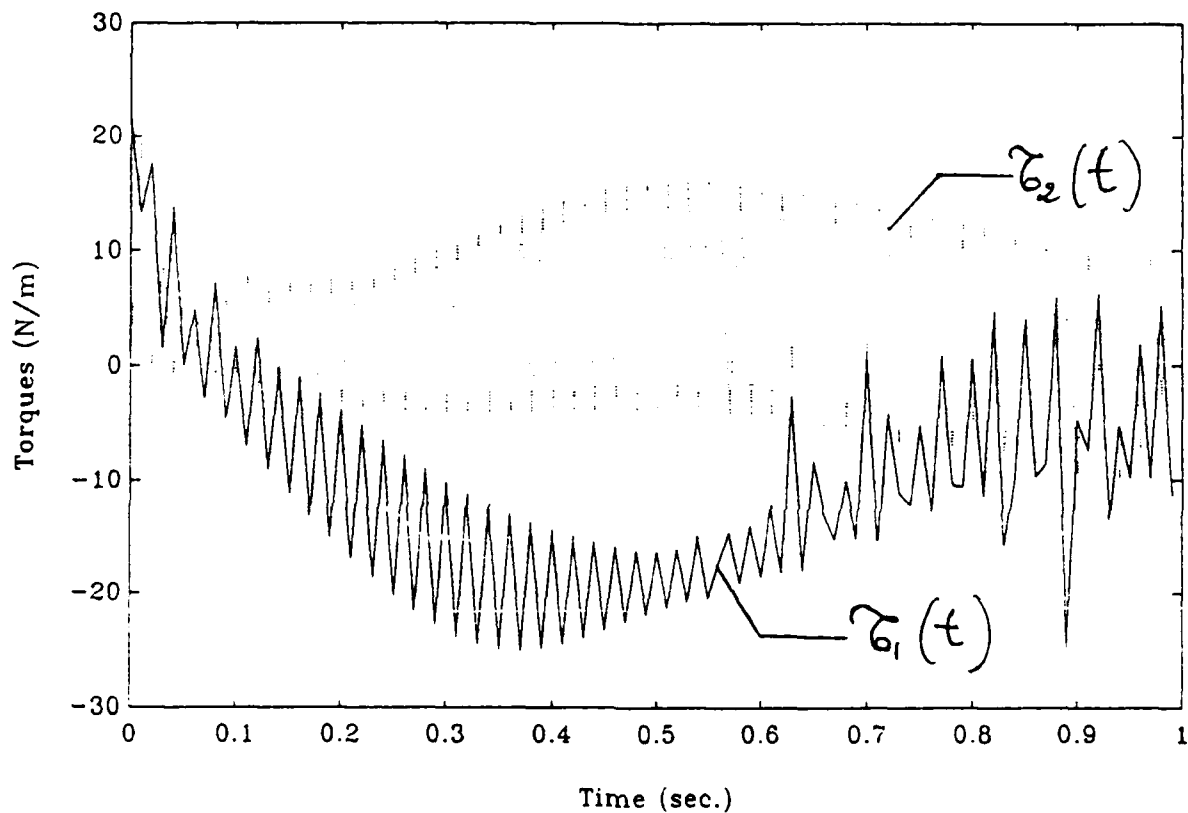


Figure 3.63: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Complete Scheme)

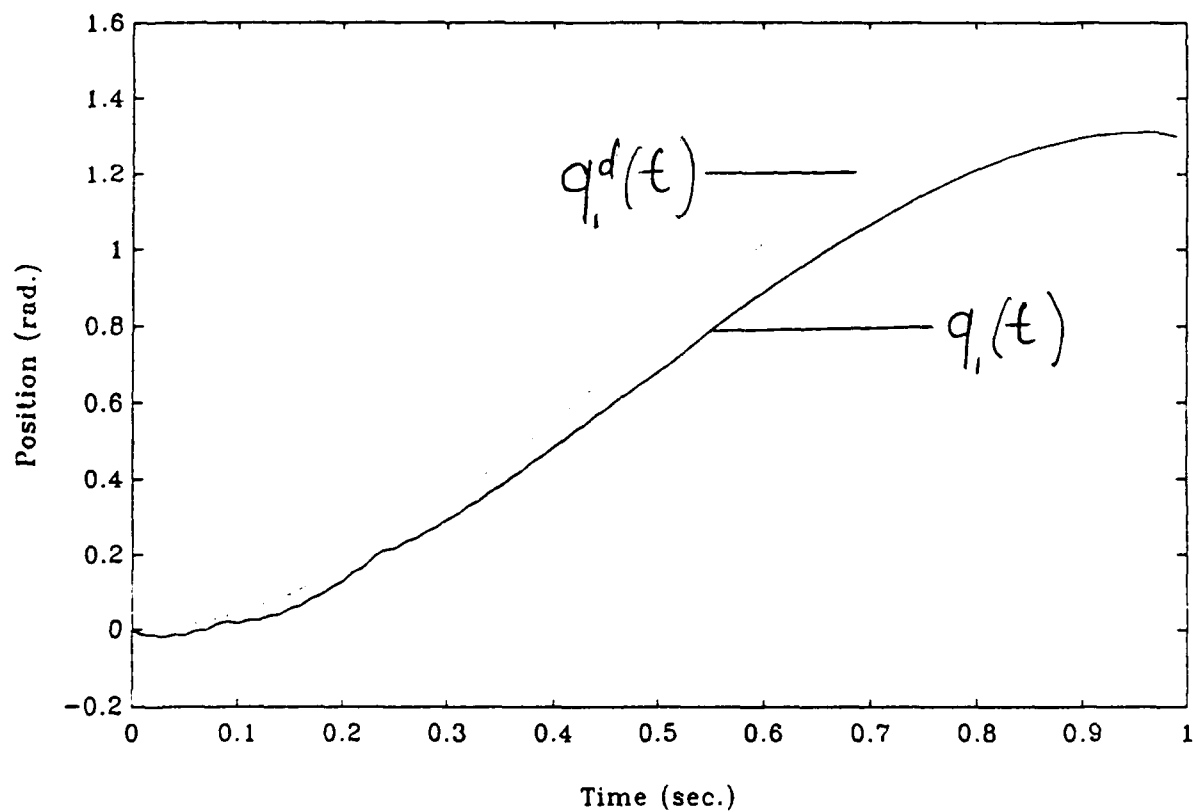


Figure 3.64: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Adaptive Perturbation Controller (With Unmodeled Disturbances)

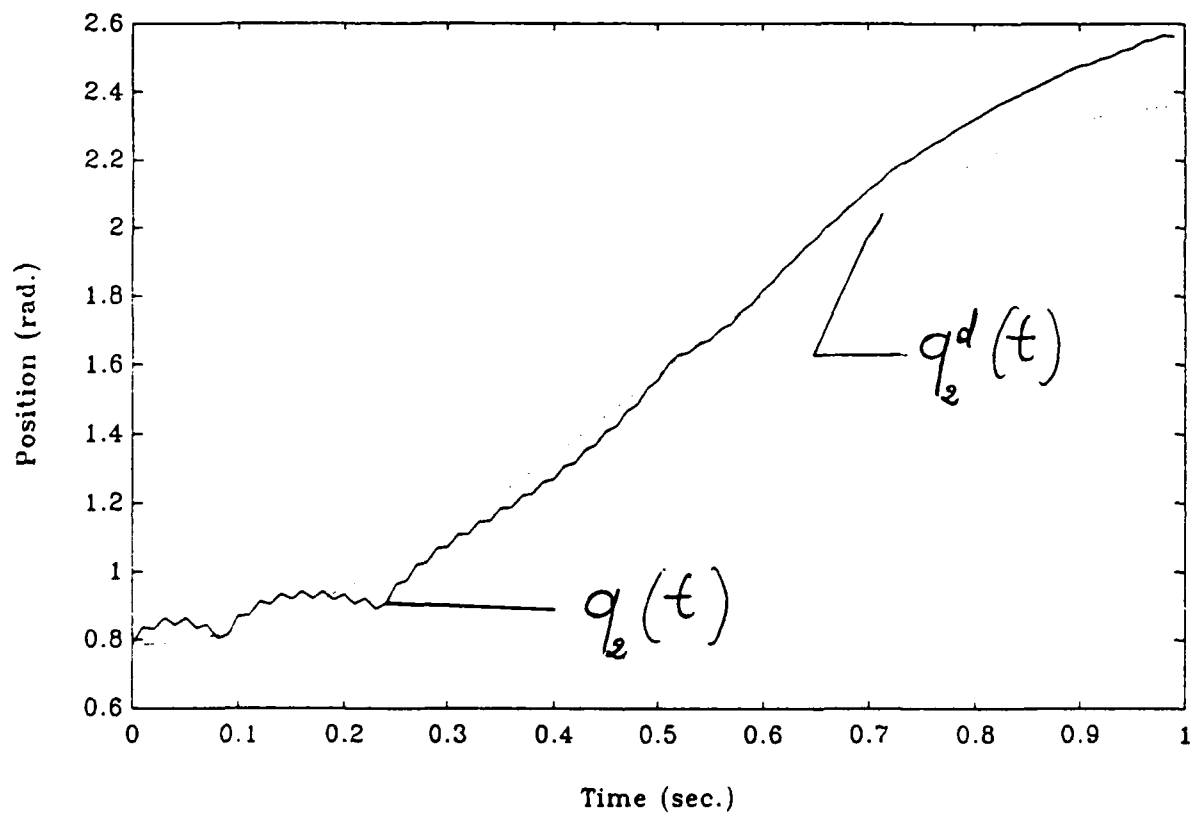


Figure 3.65: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Perturbation Controller (With Unmodeled Disturbances)

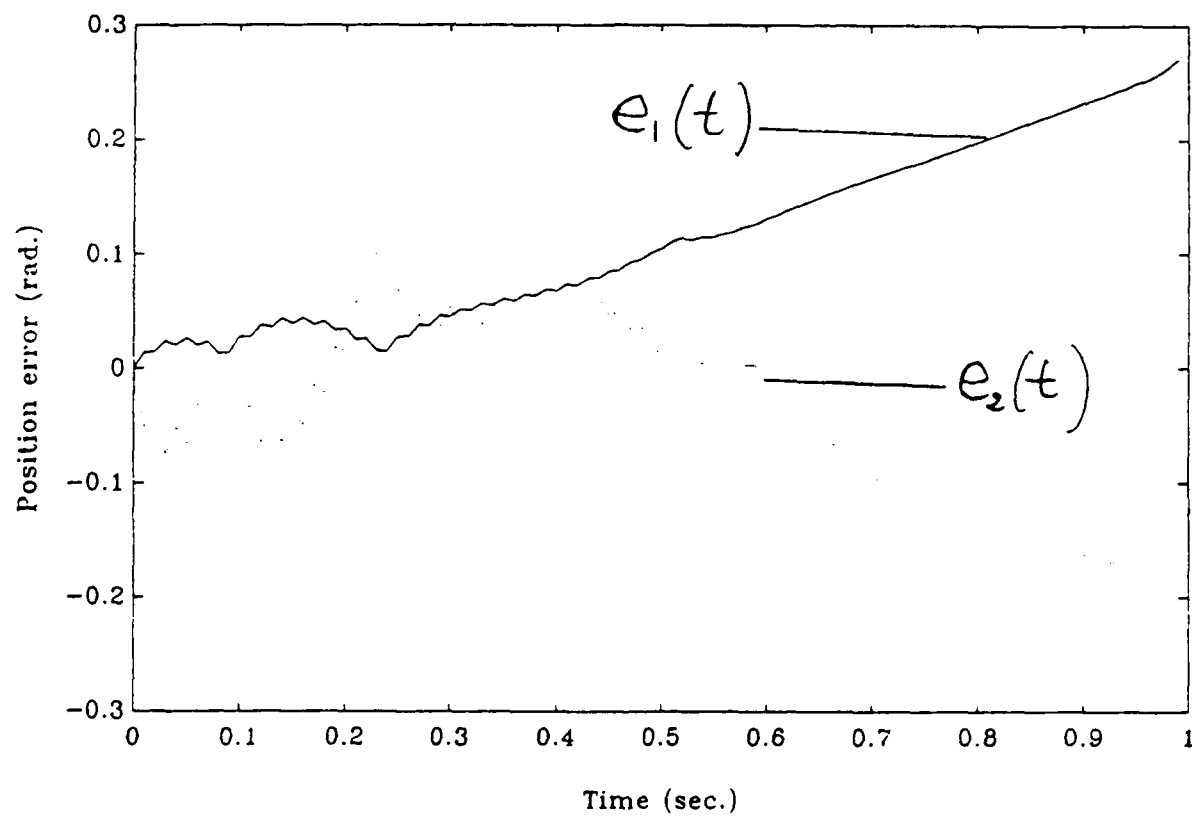


Figure 3.66: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Perturbation Controller (With Unmodeled Disturbances)

Figure 3.62). Also, as seen in Figure 3.67, the input torques are about 10 times larger than the input torques required by our control law (see Figure 3.63). Similarly, when the same conditions as above are simulated with the Linear Model Following Control Law of [63], The actual trajectories (q_1 and q_2) diverge from the desired trajectories (q_1^d and q_2^d) as seen in Figures 3.68 and 3.69. Figure 3.70 shows that the tracking errors reach, approximately, 29° in the first link and 35° in the second link. Here, again, the superiority of the proposed control methodology is evident.

A deeper insight into this adaptive control method may be gained by considering its relation to the general structure of Model Reference Adaptive Systems and to the Adaptive Model Following Controller in particular.

Referring to Figure 3.34, the same system can be represented in a slightly different, but equivalent, arrangement as shown in Figure 3.71. This particular representation highlights more clearly the parallel structure of this control law.

Now consider Figure 3.72 which gives a block diagram representation of the standard Model Following Control law [73]. A fundamental difference between Figure 3.71 and 3.72 is the fact that the flow of signals through the reference model in Figure 3.71 is reversed. This results from our formulation of the general manipulator control problem in which we assume that in practice a desired trajectory is specified by the user and not the input torques to the manipulator.

The reference model in the standard Adaptive Model Following Control is chosen to be asymptotically stable. That is, A_m is a Hurwitz matrix. The Newton-Euler recursion we are using is also a stable algorithm, in the sense that for every desired trajectory point (q^d, \dot{q}^d) where $|q_i^d| < \infty$ and $|\dot{q}_i^d| < \infty$, it yields a vector torque τ^d where $|\tau_i| < \infty$.

The feedforward matrix gain K_u , the plant feedback matrix gain K_p , and the model feedback matrix gain K_m in Figure 3.72 are chosen such that, for null initial conditions and specific plant parameter values, perfect model following exists. That is:

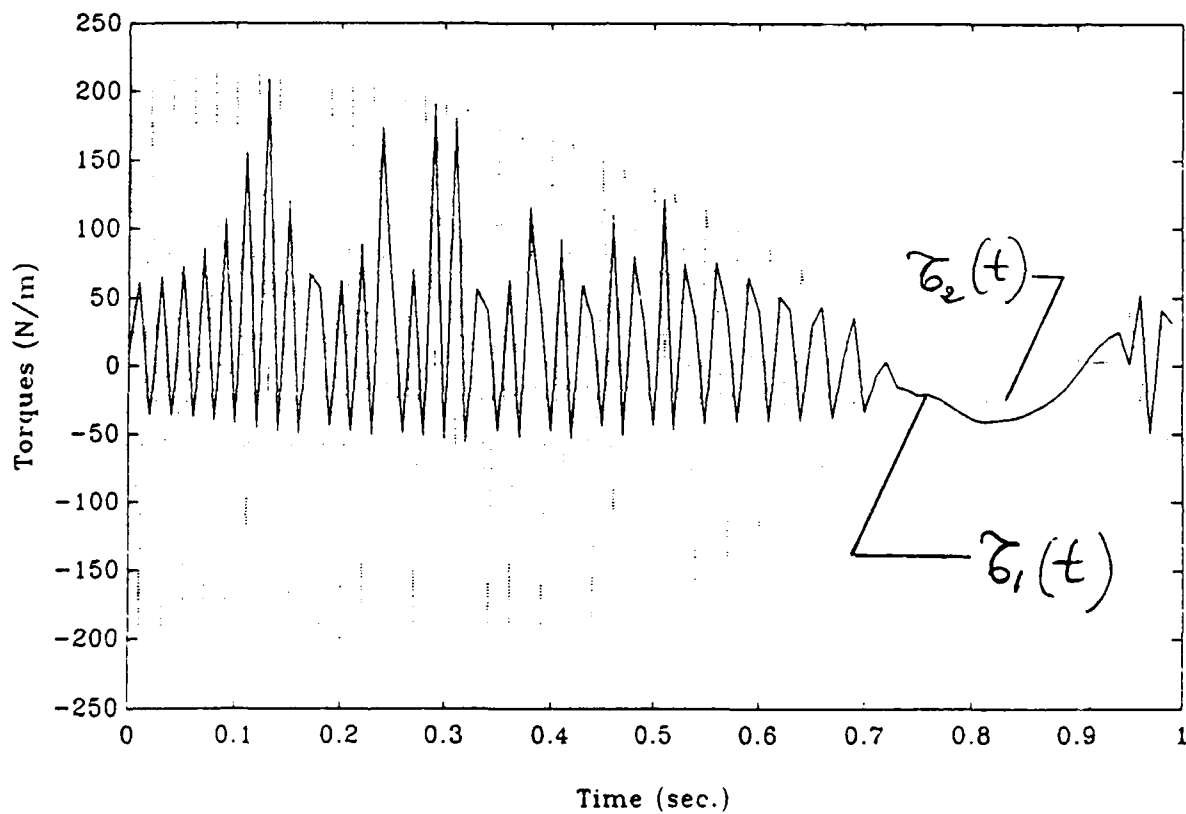


Figure 3.67: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Perturbation Controller (With Unmodeled Disturbances)

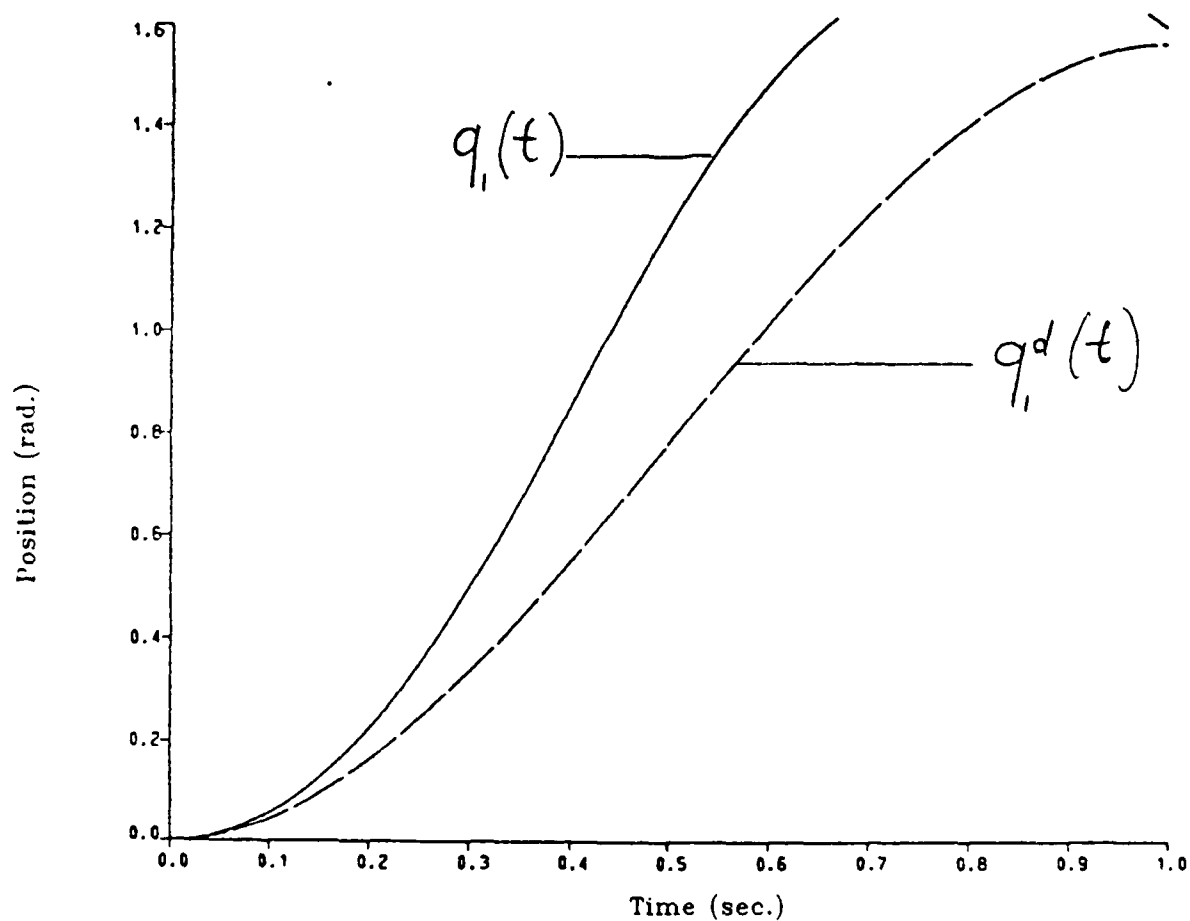


Figure 3.68: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Adaptive Model Following Controller (With Unmodeled Disturbances)

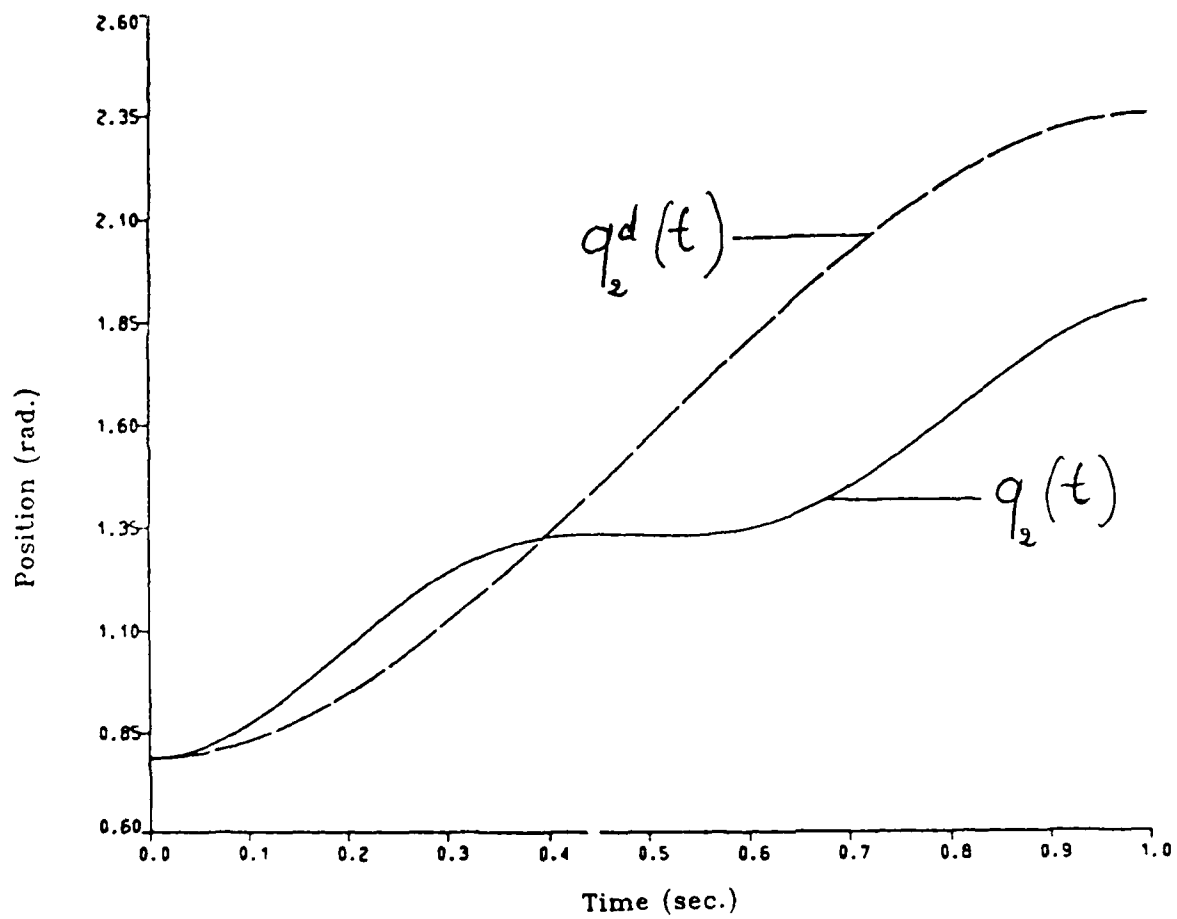


Figure 3.69: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Adaptive Model Following Controller (With Unmodeled Disturbances)

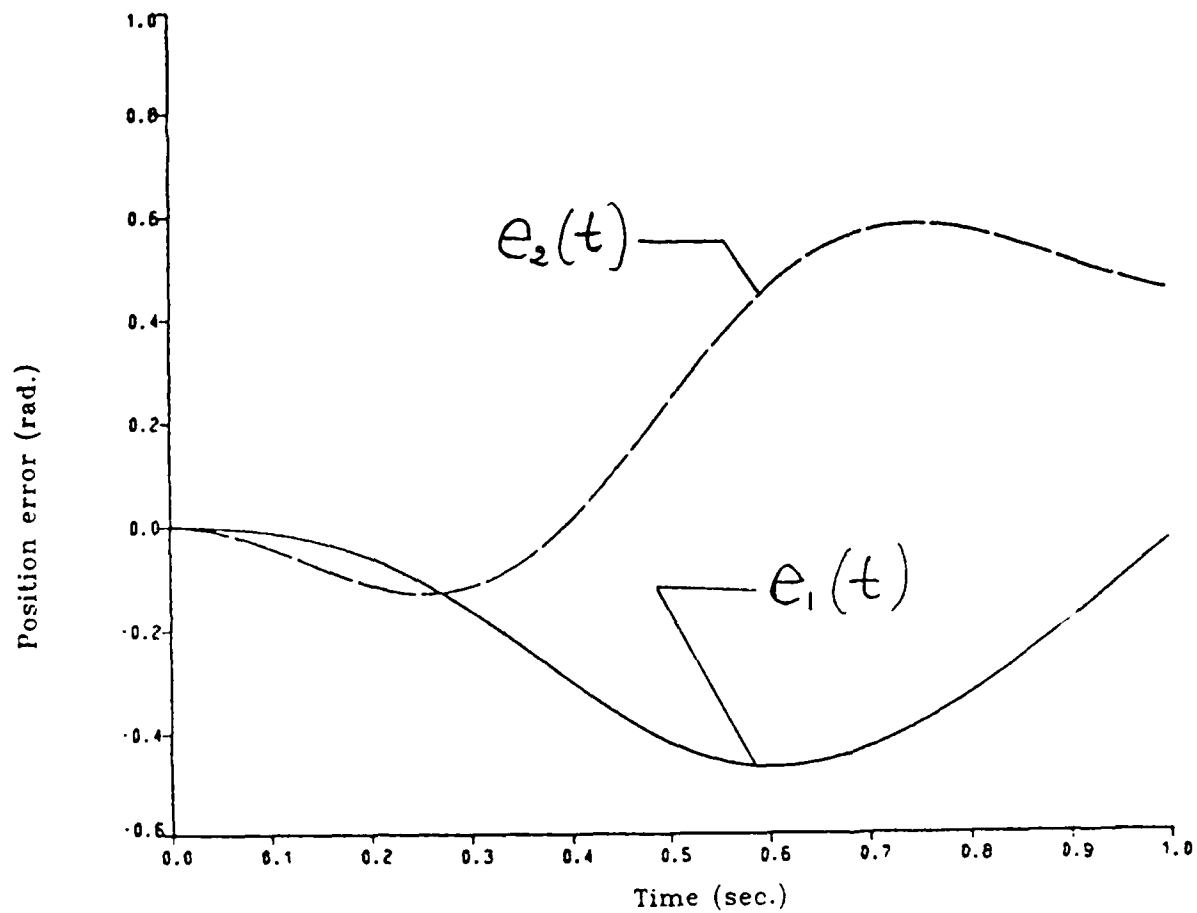


Figure 3.70: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Adaptive Model Following Controller (With Unmodeled Disturbances)

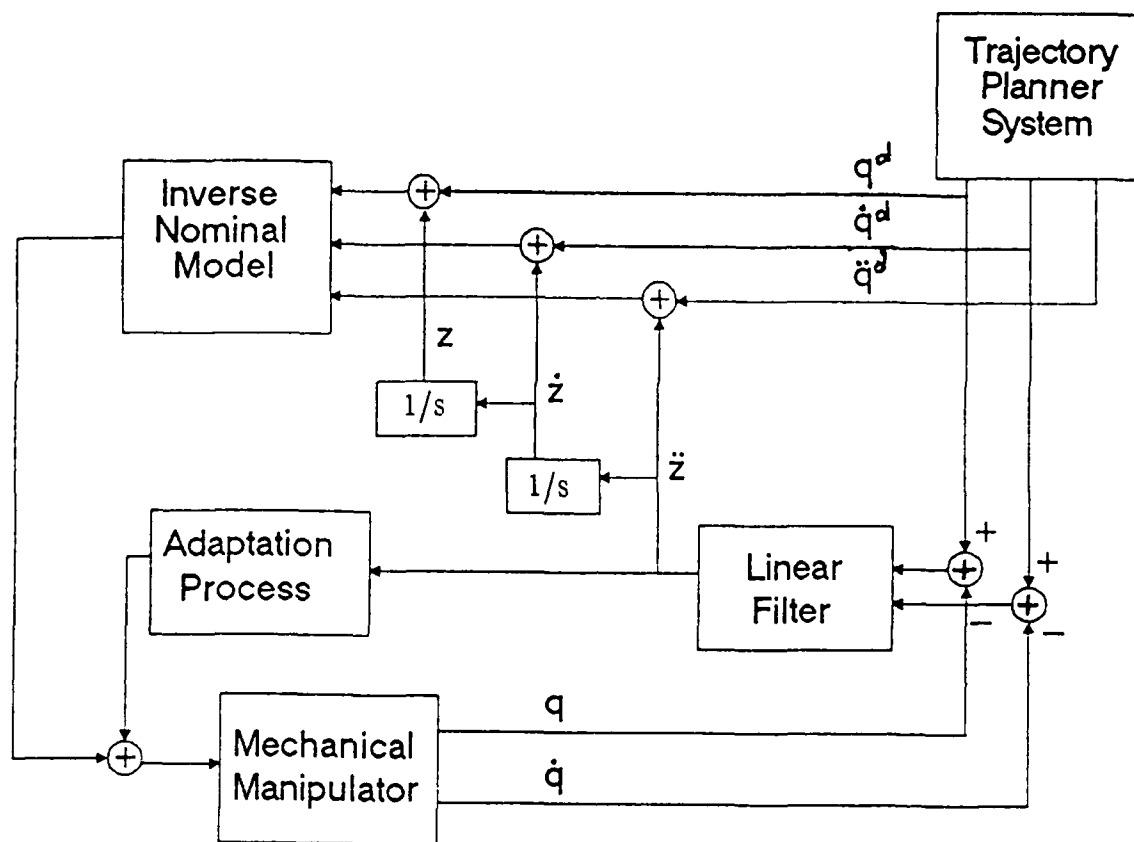


Figure 3.71: Adaptive Nonlinear Model Following Control
(Parallel Configuration)

$$\lim_{t \rightarrow \infty} A_p(t) = A_m(t) \quad (3.73)$$

$$\lim_{t \rightarrow \infty} B_p(t) = B_m(t) \quad (3.74)$$

or,

$$\lim_{t \rightarrow \infty} e(t) = 0 \quad (3.75)$$

$$\lim_{t \rightarrow \infty} \dot{e}(t) = 0 \quad (3.76)$$

In the proposed adaptive control methodology of Figure 3.71 , the reference model is an available nominal model of the plant itself used as an inverse dynamics. That is, in at least a limited range of applications for which it has been calculated, this model adequately describes the plant under consideration. When this is the case, perfect model following can be achieved without the need for any adjustment, by choosing $K_p = 0$, $K_m = 0$ and $K_u = I_n$. This particular choice of K_p , K_m , and K_u means that in the case where the values of the plant parameters are precisely known and do not vary during operation, the adaptation mechanism is not needed just as in the standard Adaptive Model Following Control. In fact, the control law reduces to an inverse dynamics control. Simulation results of this situation are shown in Figures 3.73 through 3.76. Figures 3.73 and 3.74 show that, in the ideal case where all the parameters are known, the actual trajectories (q_1 and q_2) follow very closely the desired trajectories (q_1^d and q_2^d), with no need for adaptation. The time evolution of the joint errors (e_1 and e_2) are converging to zero as we can see from Figure 3.75. Figure 3.76 shows that the corresponding input torques are within an acceptable range of values and of reasonable chattering.

As for the standard Adaptive Model Following Control, when the perfect model following exists, the role of the adaptation is to assure the convergence to this solution when the plant parameters are uncertain or vary during operation. This is shown to be the case in equations (3.64) through (3.70). This adaptation law can be classified as a signal synthesis adaptation since the feedback signals $z(t)$, $\dot{z}(t)$ and $\ddot{z}(t)$ are used to either reshape

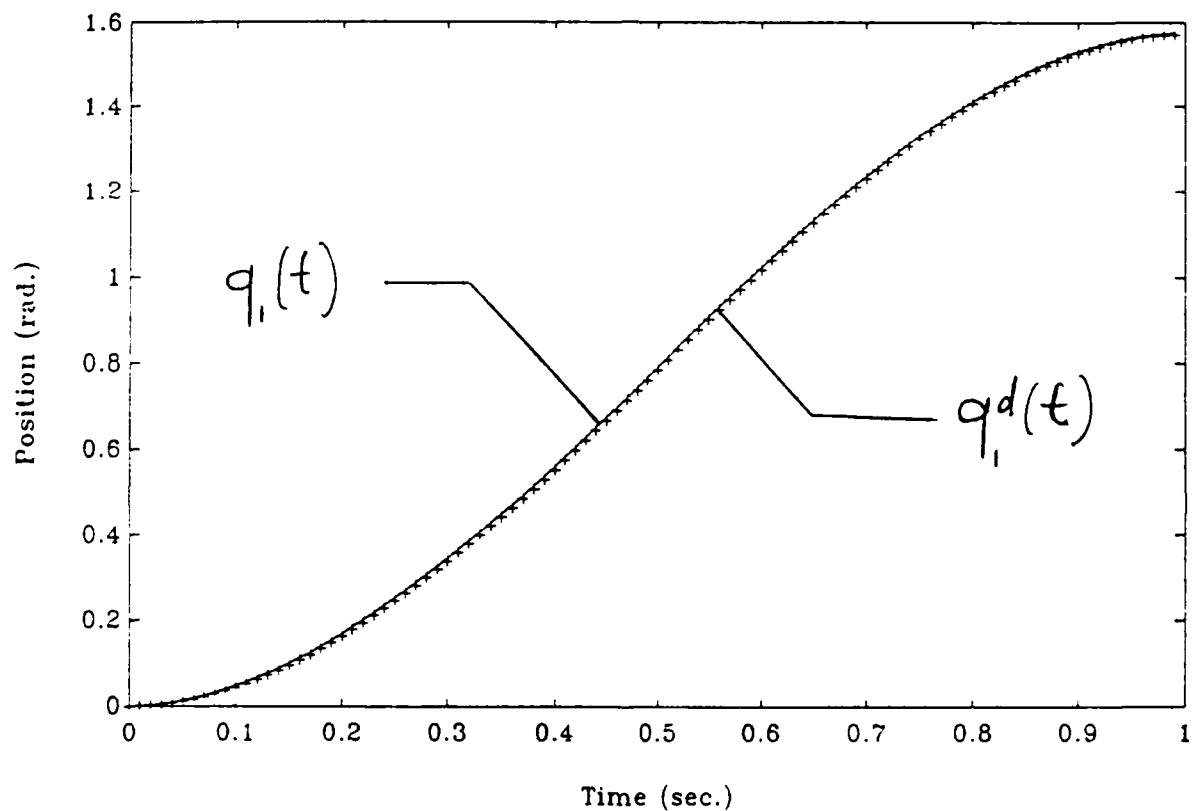


Figure 3.73: The first link desired ($q_1^d(t)$) and actual ($q_1(t)$) trajectories under the Nonlinear Adaptive Model Following Controller (Perfect Modeling)

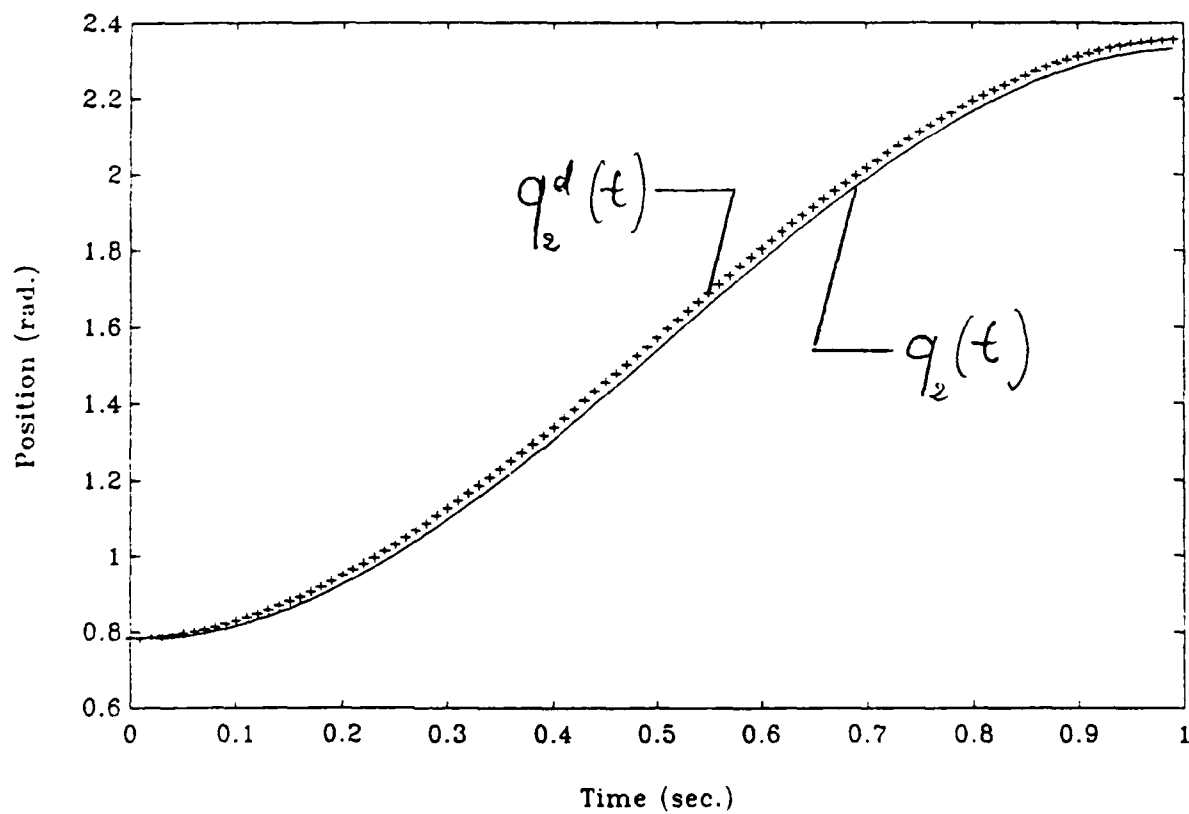


Figure 3.74: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller (Perfect Modeling)

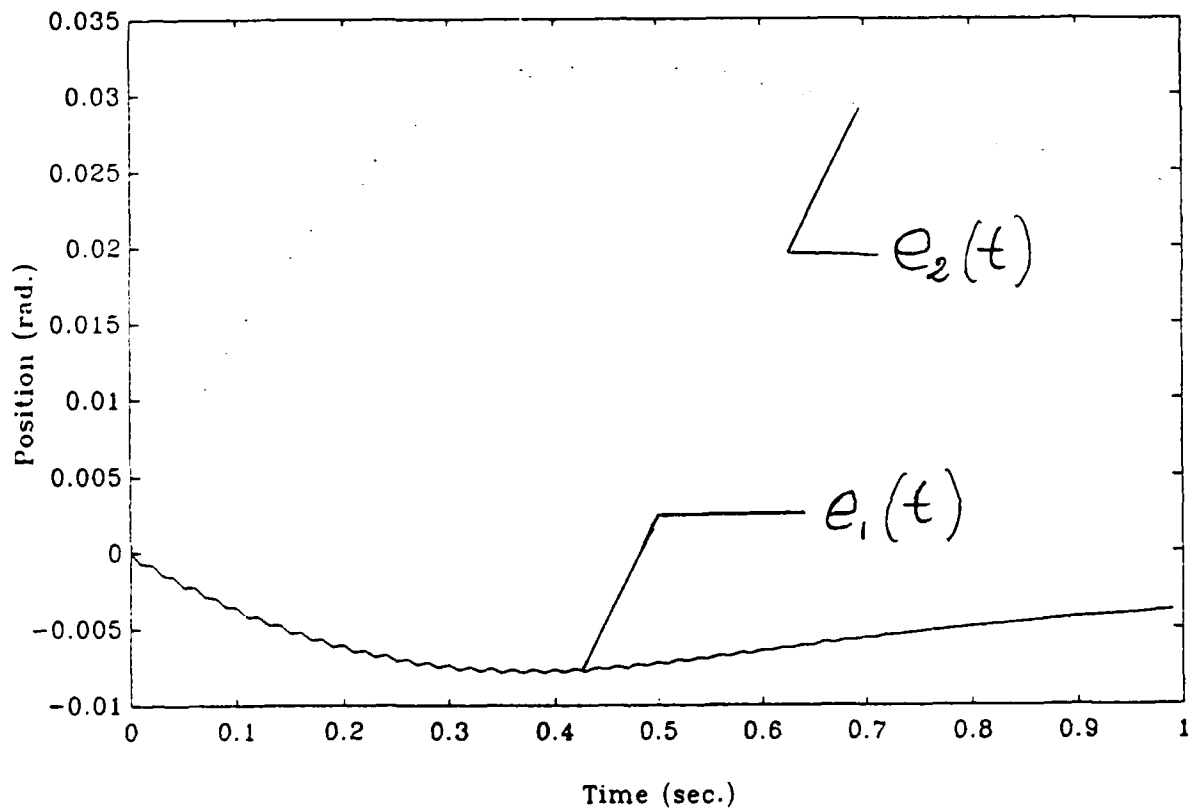


Figure 3.75: The joint one ($e_1(t)$) and the joint two ($e_2(t)$) tracking errors under the Nonlinear Adaptive Model Following Controller (Perfect Modeling)

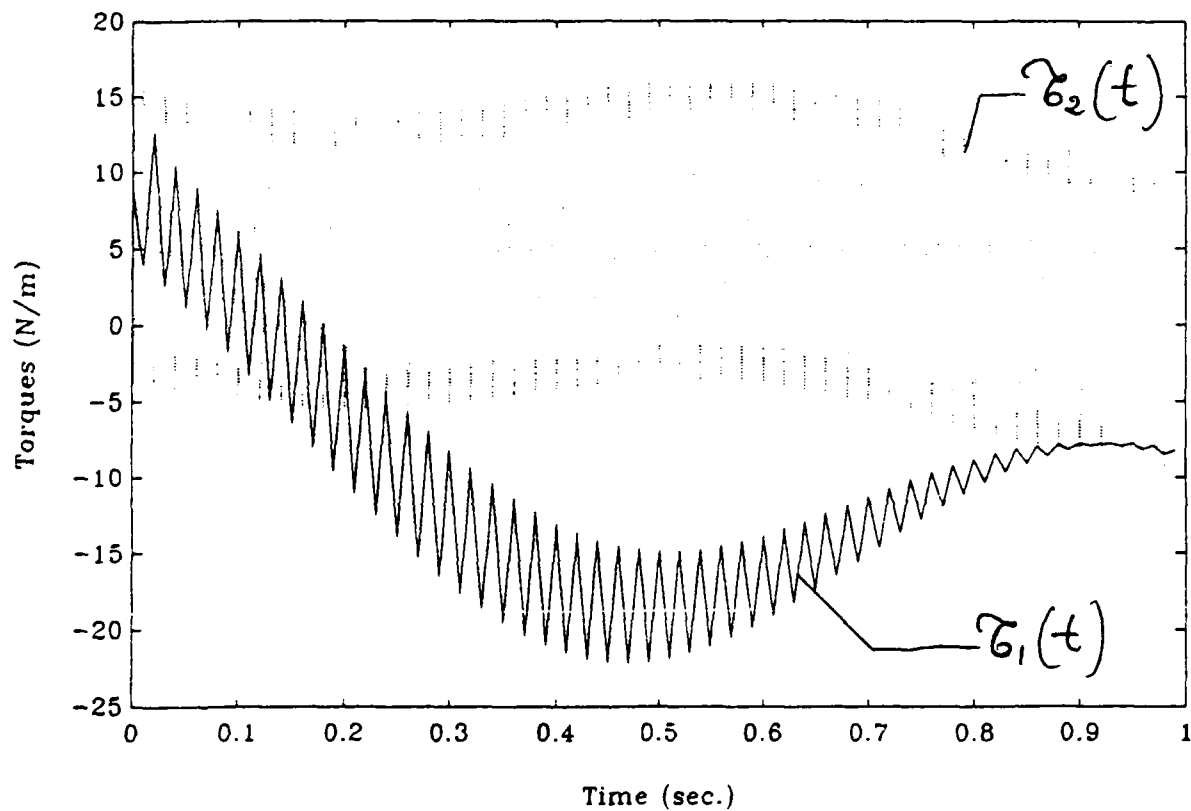


Figure 3.76: The total torques applied to joint one ($\tau_1(t)$) and to joint two ($\tau_2(t)$) under the Adaptive Model Following Controller (Perfect Modeling)

the forward torques τ_n or to generate additional torques $d\tau$ which are both acting as input signals to the plant when the values of its parameters differ from the nominal ones.

The fact that the flow of the signals through the reference model in Figure 3.71 is reversed, does not, theoretically, affect the equations governing the overall system. This can be seen from the equivalent error model representation of equation (3.57) and Figure 3.35 which is the same structure as for the standard Adaptive Model Following Control.

Finally note that this method is very insensitive to the parameters $\ell(t)$. Simulation results when

$$\ell(t) = \begin{bmatrix} 20 & 0 \\ 0 & 16 \end{bmatrix}$$

and when

$$\ell(t) = \begin{bmatrix} 120 & 0 \\ 0 & 118 \end{bmatrix}$$

are reported in Figures 3.77 through 3.80. Figures 3.77 and 3.78 are practically the same as Figures 3.60 and 3.61 obtained earlier with a different gain matrix $\ell(t)$. The robustness property is inherent to the fact that the proposed control law is a switching law as can be seen from Equation (3.60). Figures 3.79 and 3.80 show that, when the gains matrix is excessively large (8 times the nominal values), the quality of the tracking is degraded. However, the joint errors experienced in this case may still be more tolerable than the the joints errors experienced with the control law of [51] or the control law of [63].

It is also very important to point out that the proposed adaptive control law is numerically more efficient than [51] since it does not explicitly estimate the feedback model, and more efficient than [63] because it does not require the use of model matching matrices. This should be very attractive feature since adaptive control techniques suffer from computational complexity in general.

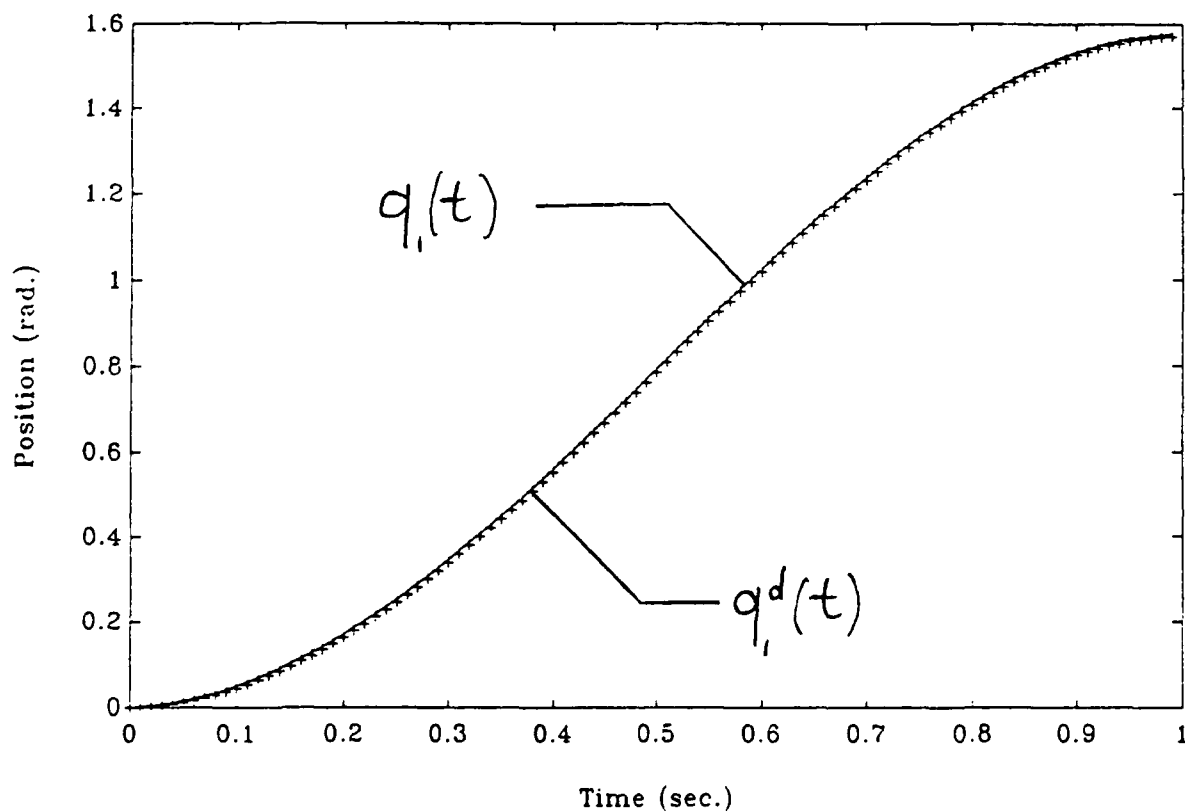


Figure 3.77: The first link desired ($q_i^d(t)$) and actual ($q_i(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (With Large Gains)

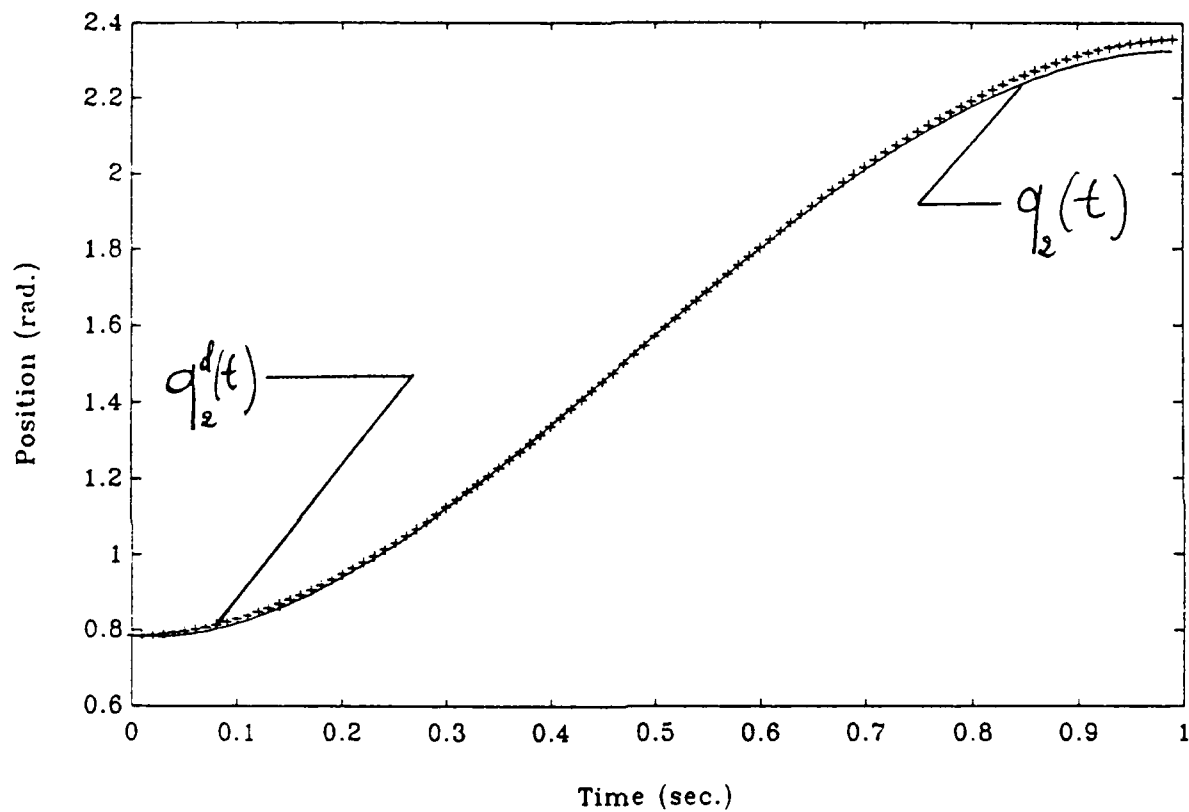


Figure 3.78: The second link desired ($q_2^d(t)$) and actual ($q_2(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (With Large Gains)

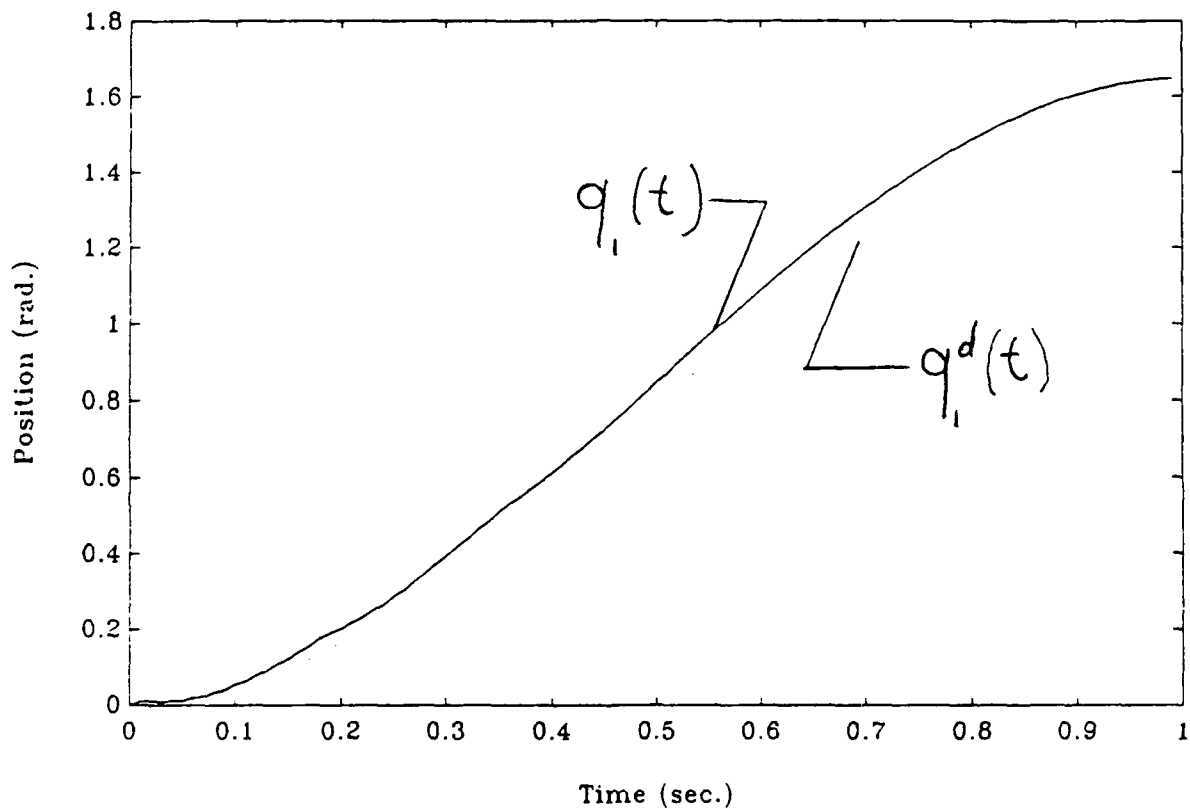


Figure 3.79: The first link desired ($q^d_1(t)$) and actual ($q_1(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Very Large Gains)

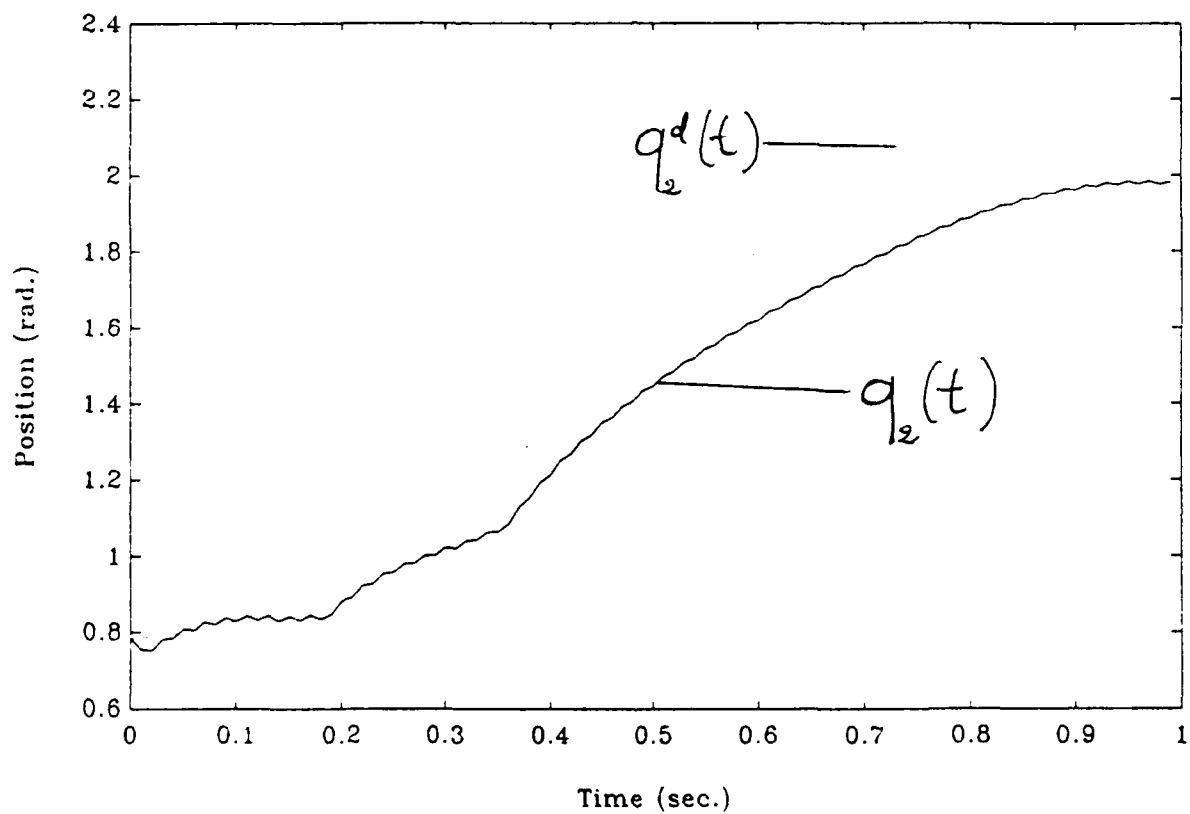


Figure 3.80: The second link desired ($q(t)$) and actual ($q(t)$) trajectories under the Nonlinear Adaptive Model Following Controller assuming Parametric and Unstructured Disturbances (Very Large Gains)

E. CONCLUSIONS AND FUTURE RESEARCH

A new adaptive control law for mechanical manipulators that maintains uniformly good performance over a wide range of motions and payloads has been developed. This control strategy has been shown to combine properties from both the Model Reference Adaptive Control and the Self Tuning Regulator theory. It has also been shown that this method serves to extend the Adaptive Model Following Control Approach into using a nonlinear model as a reference.

The design procedure is simple and systematic resulting in an overall system which is globally stable and offers itself to microprocessor implementation. We have also shown that this control law is robust with respect to variations of the plant parameters. The effectiveness of the approach has been demonstrated on several computer simulations which compare its performances against some of the commonly known adaptive control techniques. In all cases, the proposed adaptive control strategy has performed better.

This adaptive control scheme has reduced the chattering in the input torques to a "reasonable" value compared to [59] and [63]. We are currently investigating ways to eliminate this chattering completely. As has been shown in the previous section, the chattering is the result of the correction torques attempting to counterbalance the effect of errors in the manipulator parameters. If the manipulator model is precisely known, the correction torques are reduced to zero and the controller becomes an inverse dynamics. This can be achieved by off line identification tests. Also it has been shown that in practice most of the manipulator parameters can be measured or estimated beforehand and only the parameters that are load dependent are unknown [74]. Using this fact, simulations with on line recursive least squares estimation of the load alone are currently underway.

APPENDIX

DYNAMICS OF THE TWO LINKS PLANAR MECHANICAL MANIPULATOR

A. INTRODUCTION

The two revolute joints planar mechanical manipulator shown in Figure A.1 is used as the basis for our simulations throughout this study. In this Appendix, the dynamic equations describing the motion of this physical system are derived using the Lagrangian Euler equations of Chapter 2.

B. NOTATION

The same notation and conventions as established in Chapter 2 are also employed here. In addition, for $i = 1, 2$, the following variables are used to denote:

θ_i the joint angle, which also serves as the generalized coordinate;

m_i the mass of link i ;

l_i the length of link i ;

l_{ci} the distance from the proximate joint to the center of mass of link i ; and

I_i the moment of inertia of link i about the axis z_i .

C. EQUATIONS OF MOTIONS

Equation (2.15) of Chapter 2 can be used to derive the kinetic energy K_i of link i . This equation can also be broken down into a translational and a rotational parts as:

$$K_i = \frac{1}{2} m_i \mathbf{v}_{ci}^T \mathbf{v}_{ci} + \frac{1}{2} \mathbf{w}_i^T I_i \mathbf{w}_i \quad (\text{A.1})$$

where \mathbf{v}_{ci} denotes the linear velocity of the center of mass of link i and \mathbf{w}_i the angular velocity of link i about z_i .

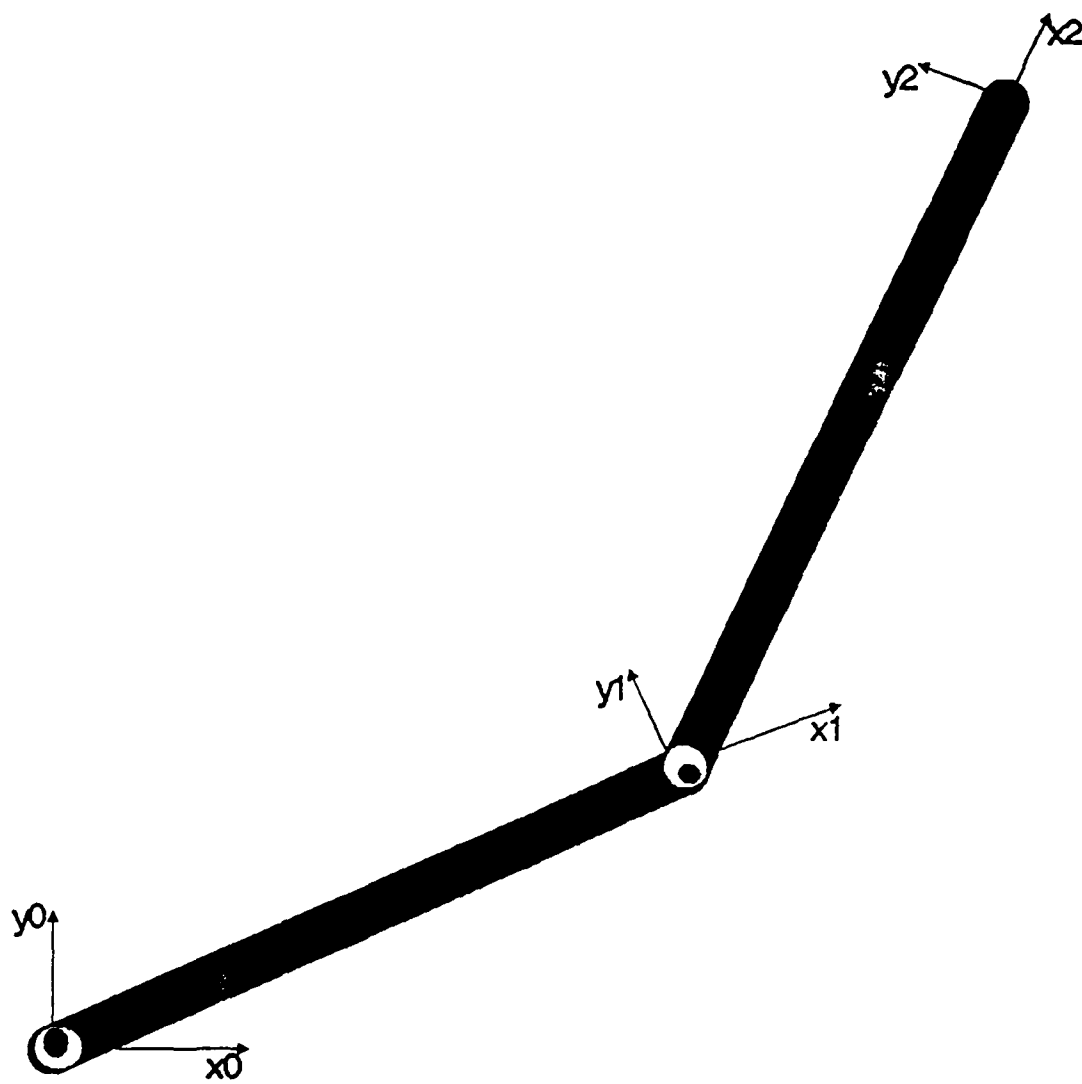


Figure A.1: A two link Mechanical Manipulator

The total kinetic energy K of the manipulator is then:

$$K = \sum_{i=1}^2 K_i \quad (\text{A.2})$$

The total potential energy is found using equation (2.20) of Chapter 2 as:

$$P = - \sum_{i=1}^2 m_i g c_i \quad (\text{A.3})$$

where c_i is the position of the center of mass of link i and g the gravity row vector.

From Figure A.1,

$$v_{c1} = \begin{bmatrix} -l_{c1} \dot{\theta}_1 \sin(\theta_1) \\ l_{c1} \dot{\theta}_1 \cos(\theta_1) \\ 0 \end{bmatrix} \quad (\text{A.4})$$

$$v_{c2} = \begin{bmatrix} -\{l_1 \sin(\theta_1) + l_{c2} \sin(\theta_1 + \theta_2)\} \dot{\theta}_1 - l_{c2} \sin(\theta_1 + \theta_2) \dot{\theta}_2 \\ \{l_1 \cos(\theta_1) + l_{c2} \cos(\theta_1 + \theta_2)\} \dot{\theta}_1 + l_{c2} \cos(\theta_1 + \theta_2) \dot{\theta}_2 \\ 0 \end{bmatrix} \quad (\text{A.5})$$

$$w_1 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix} \quad (\text{A.6})$$

and

$$w_2 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 + \dot{\theta}_2 \end{bmatrix} \quad (\text{A.7})$$

Therefore,

$$K_1 = \frac{1}{2} m_1 l_1^2 \dot{\theta}_1^2 \quad (\text{A.8})$$

$$\begin{aligned} K_2 = & \frac{1}{2} m_2 \left\{ l_1^2 + \frac{l_2^2}{4} + l_1 l_2 \cos(\theta_2) \right\} \dot{\theta}_1^2 \\ & + \frac{1}{2} m_2 \frac{l_2^2}{4} \dot{\theta}_2^2 \\ & + \frac{1}{2} m_2 l_2 \cos(\theta_1 + \theta_2) \left\{ l_1 \cos(\theta_1) + \frac{l_2}{2} \cos(\theta_1 + \theta_2) \right\} \dot{\theta}_1 \dot{\theta}_2 \\ & + \frac{1}{2} m_2 l_2 \sin(\theta_1 + \theta_2) \left\{ l_1 \sin(\theta_1) + \frac{l_2}{2} \sin(\theta_1 + \theta_2) \right\} \dot{\theta}_1 \dot{\theta}_2 \\ & + \frac{1}{24} m_2 l_2^2 (\dot{\theta}_1 + \dot{\theta}_2)^2 \end{aligned} \quad (\text{A.9})$$

$$P = (m_1 + 2m_2) g \frac{l_1}{2} \sin(\theta_1) + m_2 g \frac{l_2}{2} \sin(\theta_1 + \theta_2) \quad (A.10)$$

where l_{c1} and l_{c2} have been replaced by $\frac{l_1}{2}$ and $\frac{l_2}{2}$, respectively.

Without loss in generality, we assume $l_1 = l_2 = l$, and perform the operations in the Lagrangian equations. After rearranging, we obtain the following actuators torques:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} v_{111} & v_{112} & v_{121} & v_{122} \\ v_{211} & v_{212} & v_{221} & v_{222} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_1 \dot{\theta}_2 \\ \dot{\theta}_2 \dot{\theta}_1 \\ \dot{\theta}_2^2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \quad (A.11)$$

where

$$a_{11} = \frac{1}{3} m_1 l^2 + \frac{4}{3} m_2 l^2 + m_2 l^2 \cos(\theta_2) \quad (A.12)$$

$$a_{12} = \frac{1}{3} m_2 l^2 + \frac{1}{2} m_2 l^2 \cos(\theta_2) \quad (A.13)$$

$$a_{21} = \frac{1}{3} m_2 l^2 + \frac{1}{2} m_2 l^2 \cos(\theta_2) \quad (A.14)$$

$$a_{22} = \frac{1}{3} m_2 l^2 \quad (A.15)$$

$$v_{111} = 0 \quad (A.16)$$

$$v_{112} = -m_2 l^2 \sin(\theta_2) \quad (A.17)$$

$$v_{121} = 0 \quad (A.18)$$

$$v_{122} = -\frac{1}{2} m_2 l^2 \sin(\theta_2) \quad (A.19)$$

$$v_{211} = \frac{1}{2} m_2 l^2 \sin(\theta_2) \quad (A.20)$$

$$v_{212} = 0 \quad (A.21)$$

$$v_{221} = 0 \quad (A.22)$$

$$v_{222} = 0 \quad (A.23)$$

$$G_1 = g l \left\{ \frac{m_1}{2} \cos(\theta_1) + m_2 \left(\cos(\theta_1) + \frac{\cos(\theta_1 + \theta_2)}{2} \right) \right\} \quad (A.24)$$

$$G_2 = \frac{1}{2} m_2 g l \cos(\theta_1 + \theta_2) \quad (A.25)$$

Equations (A.11) through (A.25) constitute what is known as the inverse dynamics form of the equations of motion of the two link mechanical manipulator of Figure A.1. These

equations are used in Chapter Three to evaluate the performance of many commonly known adaptive control algorithms as applied to robotic manipulators. The reason for this choice is that Equations (A.11) through (A.25) are relatively simple enough to keep the operations manageable, and yet, representative of the systems under study.

LIST OF REFERENCES

1. G. Beni and S. Hackwood, eds., *Recent Advances in Robotics*, Wiley, New York, 1985.
2. Y. Koren, *Robotics for Engineers*, Mc Graw-Hill, Saint Louis, 1985.
3. P. Coiffet, *Robots Technology: Modeling and Control*, Wiley, New York, 1985.
4. M. Brady et al, eds., *Robot Motion: Planning and Control*, MIT Press, Cambridge, MA, 1983.
5. W. Wolovich, *Robotics: Basic Analysis and Design*, Holt, Rinehart & Winston, New York, 1985.
6. R. P. Paul, *Robot Manipulators: Mathematics, Programming and Control*, MIT Press, Cambridge, MA, 1982.
7. I. Asimov and K. A. Frenkel, *Robots, Machines in Man's Image*, Harmony Books, New York, 1985.
8. P. C. Wong and P. R. W. Hudson, "The Australian Sheep Shearing Research And Development Programme," *13 Int. Symp. Ind. Robots & Robots* 7, pp.10-56, April 7, 1983.
9. "Putting an Arm on Space," *Time Magazine*, p.81, Nov. 9, 1981.
10. R. Maus and R. Allsup, *Robotics: A Manager's Guide*, Wiley, New York, 1986.
11. P. Coiffet, *Robots Technology: Interaction with The Environment*, Wiley, New York, 1985.
12. G. J. Thaler, Notes for EC4350 (Nonlinear Systems), Naval Postgraduate School, 1988 (unpublished).
13. J. J. Craig, *Introduction to Robotics: Mechanics and Control*, Addison-Wesley, Reading, MA, 1986.
14. M. Vukobratovic, D. Stokic and N. Kircanski, *Non-Adaptive and Adaptive Control of Manipulation Robots*, Springer-Verlag, 1985.
15. C. S. G. Lee, R. C. Gonzalez and R. S. Fu, eds., *Tutorial on Robotics*, Silver Spring, MO: IEEE Computer Society Press, 1983.
16. T. C. Hsia, "Adaptive Control of Robot Manipulators—A Review," *IEEE Conference on Robotics and Automation*, San Francisco, CA, 1986.

17. J. J. Craig, *Adaptive Control of Mechanical Manipulators*, Addison-Wesley, Reading, MA, 1988.
18. K. S. Fu, R. C. Gonzalez and C. S. G. Lee, *Robotics: Control, Sensing, Vision, and Intelligence*, Mc Graw-Hill, Saint Louis, 1987.
19. J. Y. S. Luh, "Conventional Controller Design for Industrial Robots - A Tutorial," *IEEE Transactions on Systems, Man, and Cybernetics* SMC-13, pp.298-316, 1983.
20. M. E. Kahn, "The Near-Minimum Time Control of Open-Loop Articulated Kinematic Chains," Stanford Artificial Intelligence Laboratory, AIM 106, December, 1969.
21. D. E. Whitney, "Resolved Motion Rate Control of Manipulators and Human Prostheses," *IEEE Transactions on Man-Machine Systems* MMS-10, pp.47-53, 1969.
22. K. K. D. Young, "Control and Trajectory Optimization of a Robot Arm," CSL Report R-701, University of Illinois, Urbana, November, 1975.
23. M. E. Kahn and B. Roth, "The near Minimum-Time Control of Open-Loop Articulated Kinematic Chains," *J. Dynamic Systems, Meas., Control* 93, pp.164-172, 1971.
24. K.M. Vukobratovic and M.D. Stokic, "Contribution to Suboptimal Control of Manipulation Robots," *IEEE Trans. on Automatic Control*, Vol. AC-28, No. 6, June, 1983.
25. E.P. Popov et al, "Synthesis of Control System of Robots Using Dynamic Models of Manipulation Mechanisms," *Proc. of VI IFAC Symp. on Automatic Control in Space*, Erevan, USSR, 1974.
26. Y.S.J. Luh, "An Anatomy of Industrial Robots and Their Controls," *IEEE Trans. on Automatic Control*, Vol. AC-28, No 2, pp.133-152, February 1983.
27. N.G. Saridi and G.S.C. Lee, "An Approximation Theory of Optimal Control for Trainable Manipulators," *IEEE Trans on Systems, Man, and Cybernetics*, Vol. SMC-9, pp.152-159, March 1979.
28. R.P. Paul, "Modeling, Trajectory Calculation and Servoing of a Computer Controlled Arm," *Stanford Artificial Intelligence Laboratory*, A.I. Memo 177, 1972.
29. A.K. Bejczy, "Robot Arm Dynamics and Control," Technical Memorandum 33-669, Jet Propulsion Laboratory, 1974.
30. R.P. Paul, "The Mathematics of Computer Controlled Manipulator," *Proc. of JACC*, Vol. 1, pp.124-131, 1977.
31. K.A. Bejczy and R.P. Paul, "Simplified Robot Arm Dynamics for Control," *Proc. of IEEE Conf. on Automatic Control*, pp.261-262, 1981.

32. H.M. Raibert and P.K.B. Horn, "Manipulator Control Using the Configuration Space Method," *The Industrial Robot*, Vol. 5, No 2, pp.69-73, June 1978.
33. K.M. Vukobratovic and M.D. Stokic, "Significance of the Force Feedback in Realizing Movements of Extremities," *IEEE Trans. on Biomedical Engineering*, December 1980.
34. R.J. Hewit and S.J. Burdess, "Fast Dynamic Decoupled Control for Robotics Using Active Force Control," *Mechanism and Machine Theory*, Vol. 16, No 5, pp.535-542, 1981.
35. H.C. Wu and R.P. Paul, "Manipulator Compliance Based on Joint Torque Control," *Proc. 19th IEEE Conf. Decision Control*, Albuquerque, NM, Vol. 1, pp.88-94, December, 1980.
36. Y.S.J. Luh, D.W. Fisher and R.P. Paul, "Joint Torque Control by Direct Feedback for Industrial Robots," *IEEE Trans. on Automatic Control*, Vol AC-28, No 2, February, 1983.
37. E.D. Whitney, "The Mathematics of Coordinated Control of Prostheses Arms and Manipulators," *Trans. of ASME, J. Dynamics Systems, Measurement, and Control*, December, 1972.
38. Y.S.J. Luh, W.M. Walker and R.P. Paul, "Resolved-Acceleration Control of Mechanical Manipulators," *IEEE Trans. on Automatic Control*, Vol. AC-25, No. 3, pp. 468-474, June 1980.
39. H.C. Wu and R.P. Paul, "Resolved Motion Force Control of Robot Manipulator," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-12, No. 3, May-June 1982.
40. S.C.J. Yuan, "Dynamic Decoupling of a Remote Manipulator System," *IEEE Trans. on Automatic Control*, Vol. AC-23, No. 4, pp.713-717, 1978.
41. D.F. Golla, S.C. Garg and P.C. Hughes, "Linear State Feedback Control of Manipulators," *Mechanical Machine Theory* 16, pp.93-103, 1981.
42. E. Freund, "Fast Nonlinear Control with Arbitrary Pole-Placement for Industrial Robots and Manipulators," *Int. Journal Robotics Research*, 1(1), pp.65-78, 1982.
43. K.K.D. Young, "Controller Design for a Manipulator Using Theory of Variable Structure Systems," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-8, 1978.
44. K.J. Astrom, "Theory and Applications of Adaptive Control: A Survey," *Automatica*, Vol. 19, pp.471-485, 1983.
45. K.J. Astrom, "Adaptive Feedback Control," *Proc. IEEE Conference on Automatic Control*, pp. 185-217, 1987.

46. A.J. Koivo and T.H. Guo, "Control of Robotic Manipulator with Adaptive Controller," *The 20th IEEE Conference on Decision and Control*, San Diego, CA., pp. 271–276, December 16–18 1981.
47. H. Elliot, T. Depkovich, J. Kelly and B. Draper, "Nonlinear Adaptive Control of Mechanical Linkage Systems with Application to Robotics," *Proc. Automatic Control Conference*, pp. 1050–1055, 1983.
48. R. Cristi, M. Das and N.K. Loh, "Adaptive Control and Identification of a Three-Link Manipulator," Tech. Report RSD-TR4.84, Center for Robotics and Integrated Manufacturing, University of Michigan, June, 1984.
49. J.J. Craig, P. Hsu and S.S. Sastry, "Adaptive Control of Mechanical Manipulators," *IEEE International Conference on Robotics and Automation*, San Francisco, CA., 1986.
50. R.H. Middleton and G.C. Goodwin, "Adaptive Computed Torque Control for Rigid-Link Manipulators," *Proc. of 25th Conference on Decision and Control*, Athens, Grece, pp. 68–73, 1986.
51. C.S.G. Lee and M.J. Chung, "An Adaptive Control Strategy for Mechanical Manipulators," *IEEE Trans. on Automatic Control*, Vol. AC-29, pp. 837–840, 1984.
52. K.D. Vukobratovic and N.D. Kircanski, "An Approach to Adaptive Control of Robotic Manipulators," *Automatica*, Vol. 21, pp. 639–647, 1985.
53. Y.D. Landau, *Adaptive Control – the Model Reference Approach*, Marcel Dekker, Inc., 1979.
54. S. Dubowsky and D.T. Desforges, "The Application of Model Reference Adaptive Control to Robotic Manipulators," *ASME Journal of Dynamic Systems Measurement and Control*, Vol. 101, pp. 193–200, 1979.
55. S. Dubowsky, "On the Adaptive Control of Robot Manipulator – the Discrete time Case," *Proc. JACC TA-2B*, 1981.
56. S. Dubowsky, "On the Development of High Performance Adaptive Control Algorithms for Robotic Manipulators," *2nd Inter. Symp. on Robotics Research*, Kyoto, 1984.
57. M. Takegaki and S. Arimito, "An Adaptive Trajectory Control of Manipulators," *Int. Journal of Control*, Vol. 34, pp. 219–230, 1981.
58. K.K.D. Young, "Design of Variable Structure Model-Following Control Systems," *IEEE Trans. Automatic Control*, Vol. AC-23, p.1079–1085, 1978.
59. V.I. Utkin, "Variable Structure Systems with Sliding Modes," *IEEE Trans. Automatic Control*, Vol. AC-22, No. 2, pp. 212–222, 1977.

60. J.J. Slotine and S.S. Sastry, "Tracking Control of Nonlinear Systems Using Sliding Surfaces with Applications to Robot Manipulators," *Int. J. Control*, Vol. 39, No. 2, 1983.
61. R. Horwitz and M. Tomizuka, "An Adaptive Control Scheme for Mechanical Manipulators – Compensation of Nonlinearity and Decoupling Control," *Proc. of the ASME Winter Annual Meeting*, Chicago, IL, November 16–21, 1980.
62. R.P. Annex Jr. and M. Hubbard, "Modeling and Adaptive Control of a Mechanical Manipulator," *J. of Dynamic Systems, Measurement, and Control*, Vol. 106, 1984.
63. A. Balestrino, G. De Maria and L. Sciavicco, "An Adaptive Model Following Control for Robotic Manipulators," *J. of Dynamic Systems, Measurement, and Control*, Vol. 105, 1983.
64. M.E. Kahn, "The Near–Minimum–Time Control of Open–Loop Articulated Kinematic Chains," Stanford Artificial Intelligence Laboratory, AIM 106, December, 1969.
65. J.J. Uicker, "On the Dynamic Analysis of Spatial Linkages Using 4 by 4 Matrices," Ph.D. Thesis, Department of Mechanical Engineering and Astronautical Sciences, Northwestern University, 1965.
66. R.C. Waters, "Mechanical Arm Control," Artificial Intelligence Laboratory, MIT, AIM 549, October, 1979.
67. J.M. Hollerbach, "A Recursive Formulation of Lagrangian Manipulator Dynamics," *IEEE Trans. Systems, Man, Cybernetics* SMC–10, 11, pp.730–736, 1980.
68. Y.S.J. Luh, M.W. Walker and R.P. Paul, "On–line Computational Scheme for Mechanical Manipulators," *J. Dynamic Systems, Measurement, Control* 102, pp. 69–76, 1980.
69. W.M. Syn, N.N. Turner and D.G. Wyman, *DSL/360 Digital Simulation Language User's Manual*, IBM Corporation, November 1968.
70. G.H. Golub and C.F. Van Loan, *Matrix Computations*, John Hopkins University Press, 1983.
71. J.J. Slotine and W. Li, "On the Adaptive Control of Robot Manipulators," *Int. J. of Robotics Research*, Vol. 6, 1987.
72. V.I. Utkin, *Sliding Modes and their Applications*, Moscow: Mir, 1978.

INITIAL DISTRIBUTION LIST

1. Defenense Technical Information Center 2
Cameron Station
Alexandria, Virginia 22304-6145
2. Marine Royale du Maroc 4
via Embassy of the Kingdom of Morocco
Office of the Defense Attache'
1601 21st street, NW, Washington DC 20009
3. Library, Code 0142 2
Naval Postgraduate School
Monterey, California 93943-5002
4. Prof. John Powers, Chairman, Code 62 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
5. Prof. Roberto Cristi, Code 62CX 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
6. Prof. Jeff Burl, Code 62BX 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
7. Prof. Harold Titus, Code 62TS 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
8. Prof. George Thaler, Code 62TX 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000